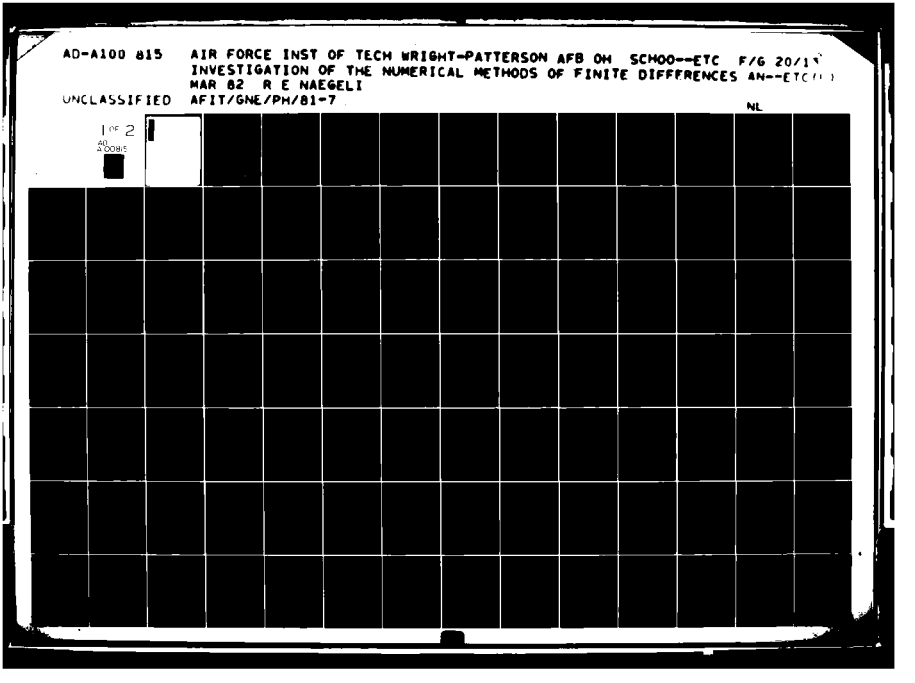
AD-A100 815
AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/G 20/1
INVESTIGATION OF THE NUMERICAL METHODS OF FINITE DIFFFRENCES AN--ETC(U)
MAR 82 R E NAEGELI
UNCLASSIFIED AFIT/GNE/PH/81-7
NL
1 OF 2
AD A100815

AFIT/GNE/PH/81-7

INVESTIGATION OF THE NUMERICAL METHODS OF FINITE DIFFERENCES AND WEIGHTED RESIDUALS FOR SOLUTION OF THE HEAT EQUATION

THESIS

AFIT/GNE/PH/81-7 Robert E Naegeli
Capt USAF

Approved for public release; distribution unlimited

AFIT/GNE/PH/81-7

INVESTIGATION OF THE NUMERICAL METHODS OF FINITE DIFFERENCES AND WEIGHTED RESIDUALS FOR SOLUTION OF THE HEAT EQUATION

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

by

Robert E. Naegeli, M.S.
Capt USAF
Graduate Nuclear Engineering

March 1981

Preface

In 1978 an article in the Journal of Computational Physics (Ref 6) evaluated numerical methods for elliptical partial differential equations. The results showed the method of weighted residuals used with finite element analysis yielded more accurate results in less computer time than finite difference methods. Dr. Kaplan of the Air Force Institute of Technology, Physics Department, and Dr. Kessler of the Air Force Materials Laboratory decided to investigate the method of weighted residuals for numerical solution of the heat equation. Specific application would eventually be to the transient heat equation for bodies of axial symmetry with the goal of reduced computer time for solution. This thesis represents a first step in that investigation by studying the method of weighted residuals applied on the whole domain of interest as a numerical method.

Thanks are due to Dr. Kaplan for his unfailing encouragement and guidance during the research quarters. Thanks are also due to Sally Lindsay who typed this manuscript. Her experience and professional approach did much to present the finished product in its best light. Last, my greatest thanks and appreciation are due to my wife and infant daughter for their personal sacrifices during a time which proved to be one of great personal adjustment and turmoil.

Robert E. Naegeli

Contents

List of Figures

List of Tables

Abstract

The method of weighted residuals applied on the whole domain for steady state and transient heat generation problems was compared to finite difference methods. The comparison consisted of the maximum absolute error from the exact solution and computer time required for solution.

One steady state and one transient heat generation problem were solved by collocation and Galerkin weighted residual methods and finite differences. The least squares weighted residual method was also used for the steady state problem. Both problems were one dimensional and had Dirichlet boundary conditions. Integrals for weighted residual methods were evaluated analytically to produce recursion relations. The transient problem was solved by the reduction to ordinary differential equations method for weighted residuals.

The Galerkin method was fastest to a given accuracy for both problems evaluated. The accuracy of Galerkin and other weighted residual methods was greater than finite differences after a point at low solution accuracy. This crossover point was typically two to three digits of accuracy. The polynomial trial functions used for weighted residual solutions exhibited a numerical instability for solutions of 10 terms and over increasing the maximum absolute error. Orthogonal collocation and weighted residuals on finite elements were recommended as alternate methods.

INVESTIGATION OF THE NUMERICAL METHODS OF FINITE DIFFERENCES AND WEIGHTED RESIDUALS FOR SOLUTION OF THE HEAT EQUATION

I Introduction

The method of finite differences has long been used for numerical solution of the heat equation and other differential equations similar to it. One example is the diffusion equation of nuclear engineering. Another method, the method of weighted residuals used with finite elements, has produced results of equal accuracy at a fraction of the computer time (Ref 6:323-350). These results were obtained for problems of structural stress which are similar to the steady state heat equation. This paper will investigate the method of weighted residuals (MWR) applied over the whole domain of the problem to see if it has similar advantages over finite differences.

The objective of this study is to compare and evaluate the finite difference method and the method of weighted residuals on the whole domain for the steady state and transient heat equations. The basis of the comparison will be accuracy of solution and computer time required. For this comparison the MWR will be used on the whole domain of interest and not finite elements. One steady state and one transient problem are solved using both methods and the results compared. Both problems are in one space dimension

of a cartesian coordinate system with Dirichlet boundary conditions.

The paper consists of chapters on the theory of MWR and finite differences, the problems chosen, results, and conclusions. The theory chapter discusses the MWR for steady state and transient problems and finite differences. In the problems chapter, the problems are discussed and the MWR and finite difference solutions are developed. The results and comparisons chapter lists the results of comparison and problems encountered. Last, the conclusion chapter gives the overall conclusions of the investigation.

II Theory

The MWR used on the whole domain of interest constructs a series function which approximately satisfies the differential equation and boundary conditions over the domain. The finite difference method, on the other hand, produces a table of values which approximate the solution to the differential equation and boundary conditions at the mesh points. Values of the solution between mesh points must be obtained by interpolation for finite differences, but are obtainable directly from the MWR series solution for any point in the domain.

The comparison between methods will be made by measuring the time to attain various accuracies. Accuracy is increased by finer mesh spacing for finite differences and more terms in the series solution for MWR. This chapter will cover the MWR and finite difference methods to be compared.

The Method of Weighted Residuals

The basic idea of the MWR is a series of complete and linearly independent trial functions which is made to satisfy the differential equation and boundary conditions (Ref 3:35). The series is substituted into the equation and boundary conditions and the error required to vanish in an average sense.

Take the case of general boundary conditions for the steady state problem:

$$LU(\bar{x}) = f(\bar{x}) \quad \text{in Region D} \tag{1}$$

$$KU(\bar{x}) = \psi(s) \quad \text{on Boundary S} \tag{2}$$

where L is the differential operator and K is the boundary condition operator. The solution, $U(\bar{x})$, is approximated as

$$U_N(\bar{x}) = \sum_{i=1}^{N} a_i \phi_i(\bar{x}) \tag{3}$$

where a_i are undetermined constants and $\phi_i(\bar{x})$ are the trial functions which make up the trial solution expansion (Ref 8: 258-261, 3:8). Substituting $U_N(\bar{x})$ for $U(\bar{x})$ in Eqs. (1) and (2) yields the residual, R:

$$LU_N(\bar{x}) - f(\bar{x}) = R \tag{4}$$

$$KU_N(\bar{x}) - \psi(s) = R \tag{5}$$

Since $U_N(\bar{x})$ is just an approximation, the equations may not balance. The error or residual of $U_N(\bar{x})$ instead of $U(\bar{x})$ in Eqs. (4) and (5) is integrated with a weight function to minimize that error:

$$\int_D w_j \left(LU_N(\bar{x}) - f(\bar{x})\right) dD = 0 \quad j=1,...N \tag{6}$$

$$\int_S w_j \left(KU_N(\bar{x}) - \psi(s)\right) dS = 0 \quad j=1,...N \tag{7}$$

The weight functions, w_j, may be determined in several ways and will be discussed later. There must be N weight functions to solve for the N undetermined constants.

The undetermined constants in $U_N(\bar{x})$ are found by solving Eqs. (6) and (7) simultaneously. Substituting the expansion for $U_N(\bar{x})$ of Eq. (3) into Eqs. (6) and (7) yields for j = 1,...N

$$\sum_{i=1}^{N} a_i \int_D w_j L\phi_i(\bar{x})\,dD - \int_D w_j f(\bar{x})\,dD = 0 \tag{8}$$

$$\sum_{i=1}^{N} a_i \int_S w_j K\phi_i(\bar{x})\,dS - \int_S w_j \psi(s)\,dS = 0 \tag{9}$$

Equating Eqs. (8) and (9) yields for j = 1,...N

$$\sum_{i=1}^{N} a_i \left[\int_D w_j L\phi_i(\bar{x})\,dD - \int_S w_j K\phi_i(\bar{x})\,dS\right] = \int_D w_j f(\bar{x})\,dD - \int_S w_j \psi(s)\,dS \tag{10}$$

Thus Eq. (10) becomes an N by N matrix problem of simultaneous equations in a_i

$$\underline{A}\,\underline{a} = \underline{b} \tag{11}$$

where $\underline{a}$ is the vector of arbitrary constants a_i, where $\underline{b}$ is the vector of integrals not involving a_i for j = 1,...N

$$\underline{b} = \int_D w_j f(\bar{x})\,dD - \int_S w_j \psi(s)\,dS \tag{12}$$

and where $\underline{A}$ is the coefficient matrix for j = 1,...N

$$\underline{A} = \sum_{i=1}^{N} \left[\int_D w_j L\phi_i(\bar{x})\,dD - \int_S w_j K\phi_i(\bar{x})\,dS\right] \tag{13}$$

Note that an underlined capital letter symbolizes a square matrix and an underlined lower case letter symbolizes a column matrix or vector. Once the system of simultaneous equations has been solved for the a_i, the approximate solution has been determined for the whole domain.

Types of Method Application. There are three ways in which MWR is generally applied. They are boundary methods, interior methods, and mixed methods (Ref 3:11).

In boundary methods the trial functions satisfy the differential equation exactly, but not the boundary conditions. MWR is then applied on the boundary conditions only. Then Eq. (4) for differential equation residual is satisfied exactly and those integrals in Eqs. (6) and (8) total zero. Last, Eq. (10) which specifies the matrix equations to solve for unknown constants becomes

$$\sum_{i=1}^{N} a_i \left[\int_S w_j K\phi_i(\bar{x})\, dS \right] = \int_S w_j \psi(s)\, dS \qquad (14)$$

For interior methods the trial functions satisfy the boundary conditions so MWR is applied on the differential equation only. Then as with boundary methods Eqs. (7) and (9) are no longer needed and Eq. (10) becomes for j = 1,...N

$$\sum_{i=1}^{N} a_i \left[\int_D w_j L\phi_i(\bar{x})\, dD \right] = \int_D w_j f(\bar{x})\, dD \qquad (15)$$

The problems studied in this paper are solved with interior MWR.

Last in mixed methods the trial functions may satisfy neither the differential equation nor boundary conditions. The trial functions may satisfy some parts of either. A boundary condition that is usually satisfied is the value of the function on the boundary (Ref 3:28-30). Eq. (10) remains the same for mixed MWR.

Types of Weighting Criteria. Several types of criteria for minimizing the residual exist. Each criteria has an associated set of weight functions which are used to integrate the residuals in MWR.

The weight function in the collocation method is the Dirac delta function (Ref 2:148):

$$w_j = \delta(\bar{x} - \bar{x}_j) \tag{16}$$

The delta function forces the value of the residual integral of Eqs. (6) and (7) to equal zero at the collocation point specified in the delta function. For a trial function expansion of N terms, N points are needed to generate N equations for simultaneous solution.

Choice of the collocation points can have a definite effect on the accuracy of the solution obtained for expansions with a low number of terms. A usual practice is to choose them evenly spaced through the region or boundary. At any event as the residual is made zero at more and more points, it presumably approaches zero throughout the region or boundary (Ref 3:9). Orthogonal collocation uses

orthogonal polynomials as trial functions and chooses the collocation points as the roots to the polynomials to improve the accuracy of low order expansions (Ref 3:97-98). Also a method called least squares-collocation uses more collocation points than constants to be determined. The residuals are squared, added, and minimized with respect to the constants (Ref 3:26-27). Least squares-collocation on the boundary has been applied as a method for steady-state and transient heat problems with arbitrary geometry and boundary conditions in two and three dimensions (Ref 16:103-108).

A second weighting criteria called the subdomain method integrates the residual over N subintervals of the domain. Then the weight function is (Ref 2:149)

$$w_i = \begin{cases} 1 & \text{in subdomain } i \\ 0 & \text{outside of subdomain } i \end{cases} \tag{17}$$

The Galerkin method corresponds to a third criteria for minimizing the residual. In the Galerkin method the trial functions in the approximate solution expansion of Eq. (4) are used as weight functions:

$$w_j = \phi_j \tag{18}$$

This amounts to making the weight functions orthogonal to the residual (Ref 2:149).

A fourth method is least squares. Here the integral of the square of the residual is minimized with respect to the solution constants (Ref 2:150)

$$\frac{\partial}{\partial a_i} \int_D R^2 \, d\bar{x} = 2 \int_D \frac{\partial R}{\partial a_i} R \, d\bar{x} = 0 \qquad (19)$$

where R is the residual, D represents the domain, and a_i are the solution constants of Eq. (4). The weight function for least squares is then the derivative of the residual with respect to one of the solution constants:

$$w_i = \frac{\partial R}{\partial a_i} \qquad (20)$$

This weight function then becomes the a_i term of the residual.

Other weight functions than these are used. In fact the weight function can be any complete set or N members of it (Ref 3:11).

The criteria used for the problems in this paper will be collocation, Galerkin, and least squares. These were the criteria used in the paper comparing FDM with the MWR applied in finite elements as mentioned before (Ref 6:323-350).

Trial Functions. The trial functions provide much of the power of MWR since they incorporate known information into the solution. Trial functions can incorporate the general symmetry of a problem or satisfy some of the boundary conditions (Ref 3:35). If the boundary conditions in a problem are of the first kind, such as $U_o(\bar{x}) = f_o(\bar{x}) = \psi(s)$, then a convenient form of trial solution is

$$U_o(\bar{x}) = f_o(\bar{x}) + \sum_{i=1}^{N} a_i \phi_i(\bar{x}) \qquad (21)$$

where the trial functions $\phi_i(\bar{x})$ are zero on the boundary and $f_o(\bar{x})$ reduces to $\psi(s)$ on the boundary. Then a problem like Eq. (1) and (2) can be transformed into an interior MWR problem. Also $f_o(\bar{x})$ which represents the boundary condition can be eliminated from the trial solution by applying the differential operator of Eq. (1) to the trial solution term by term. Then Eq. (1) becomes

$$LU_o(\bar{x}) = Lf_o(\bar{x}) + LU_N(\bar{x}) = f(\bar{x}) \tag{22}$$

where L is the differential operator, $f_o(\bar{x})$ is defined as in Eq. (21), and $U_N(\bar{x})$ is the new trial solution

$$U_N(\bar{x}) = U_o(\bar{x}) - f_o(\bar{x}) = \sum_{i=1}^{N} a_i \phi_i(\bar{x}) \tag{23}$$

Then Eqs. (1) and (2) become a problem with a new non-homogeneous part

$$LU(\bar{x}) = f(\bar{x}) - Lf_o(\bar{x}) \tag{24}$$

and a new boundary condition (Ref 3:30)

$$U(\bar{x}) = 0 \tag{25}$$

The set of functions chosen as trial functions $\phi_i(\bar{x})$ as in Eq. (23) must be complete and linearly independent to represent the solution to a boundary value problem (Ref 3: 35). One such set is the polynomials. Linearly independent and continuous polynomials have been proven complete for homogeneous and nonhomogeneous steady state heat equations

in one and two dimensions (Ref 3:355-356,359,Ref 8:263-265, 273,277). In this case the solutions evolved are uniformly convergent to the true solution. Then we can choose an $\varepsilon > 0$ and

$$|U_{true}(\bar{x}) - U_N(\bar{x})| < \varepsilon \tag{26}$$

where $U_{true}(\bar{x})$ is the true solution and $U_N(\bar{x})$ is the trial solution as in Eq. (23). The property of uniform convergence provides a test of accuracy since the absolute value of the error should decrease for trial solutions which are expansions of more terms of the set of trial functions. As more terms are taken the trial solution should represent the true solution more accurately. However, the exact choice of the trial functions will influence accuracy in low order expansions and affect the rate of convergence (Ref 3: 34-36).

Trial functions for time dependent problems such as the transient heat equation can be obtained using trial functions in the space variables which satisfy the boundary conditions multiplied by unknown functions of time:

$$U_N(\bar{x},t) = \sum_{i=1}^{N} A_i(t)\phi_i(\bar{x}) \tag{27}$$

where $A_i(t)$ is the unknown function of time (Ref 3:36). The time functions are found by applying MWR and the initial conditions as illustrated in the next section (Ref 3:44-45).

Transient Problems. The MWR solution methods developed earlier in this chapter were for steady state problems only.

For transient problems using the trial functions of Eq. (27) the solution process is different. The dependence of the trial functions on time as well as a multiplying constant must be determined. Applying the MWR to the boundary conditions and differential equation determines the time dependence. Then applying the MWR to the initial conditions with time equal to zero in the trial solution determines the constant multipliers (Ref 3:44-45). Such a treatment is called reduction to ordinary differential equations.

The determination of the time dependence of the trial solution for the transient heat equation requires solving simultaneous ordinary differential equations in time. The transient heat equation in dimensionless form with boundary and initial conditions in addition is

$$\nabla^2 U(\bar{x},t) + f(\bar{x},t) = \frac{\partial U(\bar{x},t)}{\partial t} \qquad (28)$$

where $U(\bar{x},t)$ is the dimensionless temperature, $f(\bar{x},t)$ is a heat generation term, and ∇^2 is the Laplacian. Note that heat generation depending on temperature is not considered here. If the trial function of Eq. (27) which satisfies the boundary conditions is substituted into Eq. (28), the residual, R, is

$$\sum_{i=1}^{N}\left[A_i(t)\nabla^2\phi_i(\bar{x})\right] + f(\bar{x},t) - \sum_{i=1}^{N}\left[\phi_i(\bar{x})\frac{\partial A_i(t)}{\partial t}\right] = R \qquad (29)$$

Applying a MWR weight criteria to this residual yields for $j = 1,\ldots N$

$$\sum_{i=1}^{N} A_i(t) \int_D w_j \nabla^2 \phi_i(\bar{x}) dD = \sum_{i=1}^{N} \frac{\partial}{\partial t} A_i(t) \int_D w_j \phi_i(\bar{x}) dD$$

$$- \int_D w_j f(\bar{x},t) dD \qquad (30)$$

where w_j is the weight function for the criteria used. Eq. (30) can be expressed in matrix form as

$$\underline{G}\ \underline{a} = \underline{H} \frac{d}{dt} \underline{a} - \underline{b}(t) \qquad (31)$$

where $\underline{G}$ and $\underline{H}$ are the coefficient matrices representing the integrals in Eq. (30), where $\underline{a}$ is the vector of $A_i(t)$, and where $\underline{b}(t)$ is the vector representing the last integral of Eq. (30) (Ref 4:735-736). If the inverse of $\underline{H}$ can be found, Eq. (31) can be reduced to

$$\frac{d}{dt} \underline{a} = \underline{H}^{-1}\ \underline{G}\ \underline{a} + \underline{H}^{-1}\ \underline{b}(t) \qquad (32)$$

where $\underline{H}^{-1}$ is the inverse of $\underline{H}$. Eq. (32) is a system of simultaneous, linear, ordinary differential equations for $i = 1,N$

$$\frac{dA_i(t)}{dt} = a'_{i1}A_1(t) + \ldots + a'_{iN}A_N(t) + f_i(t) \qquad (33)$$

where $A_i(t)$ is an element of vector $\underline{a}$, where $a'_{i1} \ldots a'_{iN}$ are elements of matrix $\underline{H}^{-1}\ \underline{G}$, and $f_i(t)$ are elements of vector $\underline{H}^{-1}\ \underline{b}(t)$ (Ref 9:218-219).

The $f_i(t)$ term in Eq. (33) makes it a nonhomogeneous system of first order differential equations. If the heat generation term $f(\bar{x},t)$ in Eq. (28) is a function of position only, the problem may be separated into a

steady state and a homogeneous transient problem. Substituting a new variable for temperature in Eq. (28) and its boundary and initial conditions can separate the problem:

$$T(\bar{x},t) = U(\bar{x},t) + V(\bar{x}) \tag{34}$$

where $T(\bar{x},t)$ is the new temperature variable, $U(\bar{x},t)$ is the temperature for the transient problem, and $V(\bar{x})$ is the temperature for the steady state problem (Ref 11:152-153). The steady-state problem can be solved by MWR developed earlier in this chapter. The homogeneous transient problem is then solved by the MWR reduction to ordinary differential equations outlined in this section.

For the homogeneous transient problem, Eq. (28) reduces to

$$\nabla^2 U(\bar{x},t) = \frac{\partial U(\bar{x},t)}{\partial t} \tag{35}$$

$$U(s,t) = \psi(s) \tag{36}$$

$$U(\bar{x},0) = g(\bar{x}) \tag{37}$$

where Eq. (36) represents boundary conditions which do not vary with time and Eq. (37) is the initial condition. The transient problem considered in this paper is described by Eq. (35), (36) and (37). Again substituting the trial function of Eq. (27) which satisfies the boundary conditions into Eq. (35) and applying MWR leads to a statement similar to Eq. (30) for j = 1,...N:

$$\sum_{i=1}^{N} A_i(t) \int_D w_j \nabla^2 \phi_i(\bar{x}) dD = \sum_{i=1}^{N} \frac{\partial}{\partial t} A_i(t) \int_D w_j \phi_i(\bar{x}) dD \quad (38)$$

where w_j is the weight function and $A_i(t)$ the unspecified function in time. Equation (38) expressed in matrix form is

$$\underline{G}\,\underline{a} = \underline{H}\,\frac{d\underline{a}}{dt} \quad (39)$$

where $\underline{G}$ and $\underline{H}$ are the coefficient matrices as before and $\underline{a}$ is the vector of $A_i(t)$. Multiplying by the inverse of $\underline{H}$ again yields a system of ordinary differential equations in time which are homogeneous in this case:

$$\frac{dA_i(t)}{dt} = a'_{i1}A_1(t) + \ldots + a'_{iN}A_N(t) \quad (40)$$

where $a'_{i1} \ldots a'_{iN}$ are the elements of the coefficient matrix $\underline{H}^{-1}\,\underline{G}$ as before.

The general solution to the problem of Eq. (40) is (Ref 9:220-222,294)

$$A_i(t) = C_1 a_{i1} e^{\lambda_1 t} + C_2 a_{i2} e^{\lambda_2 t} + \ldots + C_N a_{iN} e^{\lambda_N t} \quad (41)$$

where $\lambda_1 \ldots \lambda_N$ are the eigenvalues of the system of equations given in Eq. (39) or more specifically the matrix $\underline{H}^{-1}\,\underline{G}$. Also the $a_{1j} \ldots a_{Nj}$ are the components of the linearly independent eigenvectors of matrix $\underline{H}^{-1}\,\underline{G}$ and $C_1 \ldots C_N$ are the unspecified multipliers which multiply all elements of that eigenvector. Eq. (39) is then an eigenvalue problem since the derivative of an eigenvector $a_{ij}e^{\lambda_j t}$ is $a_{ij}\lambda_j e^{\lambda_j t}$ (Ref 9:294):

$$\underline{G}\,\underline{a} = \lambda\,\underline{H}\,\underline{a} \tag{42}$$

Forming the solution for $A_i(t)$ is then a two-step process. First the eigenvalues and eigenvectors of the system of ordinary differential equations describing the problem are found to determine the time dependence of exponential decay of the transient solution. The last step of determining the constant multipliers, $C_1 \ldots C_N$, is analogous to the MWR for steady state problems. The trial solution for time zero is substituted into the initial condition, Eq. (37), to form a residual. The residual is then minimized with a weight function to yield simultaneous equations for $C_1 \ldots C_N$ with $j = 1,\ldots N$:

$$\sum_{i=1}^{N} A_i(0) \int_D w_j \phi_i(\bar{x})\, dD = \int_D w_j g(\bar{x})\, dD \tag{43}$$

where $A_i(0)$ is the time function of Eq. (27) evaluated at the initial time, where $g(\bar{x})$ is the initial temperature distribution, w_j the weight function, and $\phi_i(\bar{x})$ the trial functions. Since $A_i(t)$ is a sum of exponentials in time, for the initial time zero they take a specific value of one. Each $A_i(0)$ then becomes a sum of the eigenvector multipliers, $C_1 \ldots C_N$ and the summation can be changed to add the C_1 terms from all $A_i(0)$ and the same for the other multipliers. The matrix system then reduces to

$$\underline{J}\,\underline{c} = \underline{d} \tag{44}$$

where $\underline{J}$ is the coefficient matrix of multipliers, $\underline{c}$ is the multiplier vector, and $\underline{d}$ is the vector of initial values. Once the simultaneous equations are solved, the $A_i(t)$ and MWR solution is finished (Ref 9:294).

Only two of the three MWR criteria used for steady state problems are applicable to this transient method. Collocation and Galerkin both result in the eigenvalue problem of Eq. (42). However, least square does not since the resulting equations have factors of λ^2 as well as λ. Only the collocation and Galerkin criterias will be used for the transient problem (Ref 2:313-314).

Numerical Solution for MWR

This section will outline the solution algorithms for the steady state and transient problems.

Steady State. The numerical solution of MWR on the whole domain of interest has been largely limited to collocation methods. These methods include least squares-collocation and orthogonal collocation mentioned earlier under the section on types of criteria. Collocation offers an advantage since no integrals need to be evaluated. Only the values of residuals at collocation points need be evaluated.

Collocation, Galerkin, and least squares criterias were used to compare MWR on finite elements with finite differences in an article by Houstis et. al.(Ref 6:323-350). In order to compare the results in this paper with the

results of Houstis, all three criteria will be used for the steady state problem.

Evaluation of the integrals for the Galerkin and least squares criteria presents another difficulty. A form of numerical integration could be used to evaluate the integrals over the domain of interest. However, since the trial functions used are simple polynomials and the integration on one dimension only for the problems in this paper, the integrals can be easily evaluated analytically to develop a recursion relation for the matrix elements. The recursion relation then needs to be evaluated only once for each matrix element where the integrand would need to be evaluated several times for numerical integration.

Another choice to be made is the choice of a matrix equation solver. Since every element of the MWR coefficient matrix will have a non-zero value, the best choice appears to be Gaussian elimination as a matrix equation solver. Iterative methods can be faster than Gaussian elimination for sparse matrices, but the rate of convergence is uncertain. The Gaussian elimination will be used for both MWR and finite difference solutions.

Then the algorithm for solution of the steady state problem for MWR consists of two steps. First, the solution matrices are formed from recursion relations. Then the matrix systems are solved to obtain the defining constants for MWR.

After the solution is obtained it is compared against the true analytical solution to determine its accuracy. Two measures of accuracy will be used. The first is absolute accuracy which should decrease as the MWR solution contains more terms. The second is fractional or percentage accuracy which is most of interest in a practical sense. The fractional error determines the number of significant figures in the answer.

After the solutions are obtained and the accuracy is checked for solutions with different numbers of terms, the time to compute the solution is measured. The central processor time of the Control Data Corporation Cyber 74/Cyber 750 computer system with NOS/BE operating system was used to measure the solution formation time. Since the central processor time output is only accurate to .01 seconds, the solution for a given number of terms must be repeated several times for accurate measurement (Ref 5:8-9).

Transient. Numerical solution of the transient MWR problem involves solution of an eigenvalue problem and a system of simultaneous linear equations. Solution of the simultaneous equations will be by Gaussian elimination as for the steady state problem. The eigenvalue problem will be solved in its general form.

The eigenvalue problem is shown in Eq. (42) which is repeated as Eq. (45) here

$$\underline{G}\,\underline{a} = \lambda\,\underline{H}\,\underline{a} \tag{45}$$

where $\underline{H}$ is the coefficient matrix for integrals which minimize the residual terms involving the time derivative, where $\underline{G}$ is the coefficient matrix for the integrals which minimize the space derivative, where $\underline{a}$ is the vector of unknown time functions, $A_i(t)$, in the trial solution, and where λ is the eigenvalue. The vector $\underline{a}$ is the set of the N linearly independent eigenvectors of the system and λ represents the N eigenvalues associated with the eigenvectors. The total solution for $A_i(t)$ is shown in Eq. (41). A subroutine which solves the eigenvalue problem without inverting the $\underline{H}$ matrix was selected from the IMSL program library. This subroutine, EIGZF, allows the $\underline{G}$ and $\underline{H}$ matrices to be input directly without inversion (Ref 10:241-256,7:EIGZF1-5).

The algorithm for computing the transient MWR solution involves four steps. First the $\underline{G}$ and $\underline{H}$ matrices are computed from Eq. (38). Next the eigenvalues and eigenvectors are found by EIGZF subroutine. Then the coefficient matrices for calculation of the constant multipliers are evaluated as indicated in Eq. (43) and (44). Last the system is solved for the multipliers.

As for the steady state problem, the accuracy of solution and formation time of solution must be measured. The one MWR solution is good for all times due to the exponential time dependence of the time function $A_i(t)$ of Eq. (27). Time for solution must be measured for different numbers of terms in the MWR solution. Then accuracy at different times is measured against the analytical solution.

The Finite Difference Method

The finite difference method approximates the derivatives of a differential equation by the ratios of differences between points in the domain of the problem. The system of equations that results can be solved for the values of the function in the differential equation at the mesh points. Since the finite differences only approximate the derivatives, the values for the function have some error based on the mesh spacing.

The two derivatives in the steady state and transient heat equations will be approximated by central and forward differences. The two derivatives are the second derivative with respect to the space variable and the first derivative with respect to time:

$$\frac{\partial^2 U(x,t)}{\partial x^2} = \frac{\delta^2 U(x,t)}{h^2} = \frac{U(x+h,t) - 2U(x,t) + U(x-h,t)}{h^2} \tag{46}$$

$$\frac{\partial U(x,t)}{\partial t} = \frac{\Delta U(x,t)}{k} = \frac{U(x,t+k) - U(x,t)}{k} \tag{47}$$

where Eq. (46) shows the second derivative with respect to x, Eq. (47) shows the first derivative with respect to time, where U(x,t) is the function differentiated, where h and k are the distances between mesh points in x and time respectively, where δ denotes a central difference taken about the point (x,t) where the derivative is approximated, and where Δ denotes a forward difference taken from the point (x,t)

(Ref 15:59-60). Eq. (46) and (47) are also valid for ordinary derivatives.

For the steady state problem the unknown value of the function at all mesh points will be found simultaneously. The differential equation using finite difference approximations is formulated for each mesh point. Then the system of equations is solved simultaneously by the same Gaussian elimination method used for MWR.

Explicit methods will be used for the transient problem. In explicit methods, the differential equation with finite difference approximations is used to compute the value at a next point from values at known points.

Since the difference expressions of Eqs. (46) and (47) are derived by combining Taylor series expansions of the function to be differentiated, the error in the expressions can be found by examining the expansion terms not used in the expression. The central difference then has an error proportional to h^2 or of order h^2, $O(h^2)$, where h is the mesh spacing. The first forward difference has an error O(k) where k is the mesh spacing. The mesh spacing determines the error then and closer mesh spacing will give more accurate results. Also note that for an equation that uses two difference expressions, such as the transient heat equation, the error is the sum of the two errors (Ref 15:59-60, 108).

III The Problem Set

One steady state and one transient heat problem are solved by the MWR on the whole domain and by the finite difference method. Both problems are in one space dimension with Dirichlet boundary conditions. This chapter discusses the problems and their solutions. To prevent any advantage between methods all MWR and finite difference solutions were programmed in Fortran 4 Extended. The programs were also compiled with the same compilation option, option 1.

Steady State Problem

For the steady state problem consider for $0 \leq x \leq 1$

$$\frac{d^2U(x)}{dx^2} + U(x) + x = 0 \tag{48}$$

$$U(0) = U(1) = 0 \tag{49}$$

where Eq. (48) represents the heat equation, where Eq. (49) is the homogeneous Dirichlet boundary condition, where U(x) is temperature, and where U(x)+x represents a heat generation term. Removal of the thermal conductivity constant factor which is normally shown multiplying the second derivative of temperature and reduction of the equation to this form requires a very special heat generation term. However, study of the equation in this form provides a convenient closed form solution (Ref 8:269):

$$U(x) = \frac{\sin(x)}{\sin(1)} - x \tag{50}$$

For the MWR solution to the steady state problem a polynomial trial function which satisfies the boundary conditions and has been proven complete for similar problems is (Ref 3: 356)

$$\phi_i = x^i(1-x) = x^i - x^{i+1} \tag{51}$$

Then the trial solution $U_N(x)$ will be

$$U_N(x) = \sum_{i=1}^{N} a_i(x^i - x^{i+1}) \tag{52}$$

where a_i are undetermined constants. Substituting Eq. (52) into Eq. (48) yields the residual, R, of the differential equation for this interior method application of MWR:

$$R = \sum_{i=1}^{N} a_i[i(i-1)x^{i-2} - i(i+1)x^{i-1} + x^i - x^{i+1}] + x \tag{53}$$

Simultaneous equations can now be constructed from the residual by integration with the weight functions of the three MWR methods to be evaluated, for j = 1,...N:

$$\sum_{i=1}^{N} a_i \int_0^1 w_j[i(i-1)x^{i-2} - i(i+1)x^{i-1} + x^i - x^{i+1}]dx$$

$$= -\int_0^1 w_j \, x \, dx \tag{54}$$

where w_j is the weight function and Eq. (54) represents one of the N simultaneous equations. Using the weight functions

of Eq. (15), (16), and (17) the integrals are evaluated to give recursion relations. The recursion relations evaluated for values of i and j provide the matrix elements for the matrix problem $\underline{A}\ \underline{a} = \underline{b}$ where $\underline{A}$ is the coefficient matrix defined by the left side of Eq. (54), $\underline{b}$ is the vector defined by the right side of Eq. (54), and where $\underline{a}$ is the vector of undetermined constants, a_i. The specific weight function, w_j, used defines the rows of $\underline{A}$ and the ith term of the trial solution defines the columns. Once the matrix problem has been solved for $\underline{a}$ the MWR solution is finished.

Recursion relations for the collocation method are the residuals:

$$A(j,i) = i(i+1)x_j^{i-1} - i(i-1)x_j^{i-2} + x_j^{i+1} - x_j^i \qquad (55)$$

$$b(j) = x_j \qquad (56)$$

where x_j is the jth of the N equally spaced collocation points, A(j,i) is an element of the matrix $\underline{A}$, and b(j) is an element of vector $\underline{b}$. Also the recursion relations for the Galerkin method using the trial functions as weight functions are

$$A(j,i) = \frac{[(i+1)i + i(i-1)]}{(j+i)} + \frac{2}{(j+i+2)} - \frac{1}{(j+i+3)} - \frac{i(i-1)}{(j+i-1)} - \frac{i(i+1) + 1}{(j+i+1)} \qquad (57)$$

$$b(j) = \frac{1}{(j+2)(j+3)} \qquad (58)$$

Last, the recursion relations for the least squares method with terms of the residual as weight functions are

$$A(j,i) = \frac{j(j-1)i(i-1)}{(j+i-3)} - \frac{[j(j-1)i(i+1) + i(i-1)j(j+1)]}{(i+j-2)}$$

$$+ \frac{[j(j-1) + j(j+1)i(i+1) + i(i-1)]}{(i+j-1)}$$

$$- \frac{[j(j-1) + j(j+1) + i(i+1) + i(i-1)]}{(i+j)}$$

$$+ \frac{[i(i+1) + 1 + j(j+1)]}{(i+j+1)} - \frac{2}{(i+j+2)} + \frac{1}{(i+j+3)} \tag{59}$$

$$b(j) = 1 - \frac{1}{(j+2)(j+3)} \tag{60}$$

The $\underline{A}$ matrices resulting from the three methods have all non-zero elements.

The finite difference solution is constructed using a central difference approximation for the second derivative of temperature with respect to x. The central difference approximation of Eq. (46) for temperature as a function of x only is

$$U_j^{11} = \frac{U_{j+1} - 2U_j + U_{j-1}}{h^2} \tag{61}$$

where the subscript j identifies the particular mesh point of N equally spaced mesh points, where U_j^{11} is the second derivative with respect to x, and where h is the spacing between mesh points. Then Eq. (48), the differential equation, becomes the difference equation

$$- U_{j-1} + (2-h^2)U_j - U_{j+1} = h^2 x_j \qquad (62)$$

where Eq. (61) has been substituted into Eq. (48) and multiplied by $-h^2$. With N mesh points the interval 0 to 1 of the problem is divided into N+1 spaces so $h = 1/(N+1)$. Then the matrix problem $\underline{A}\ \underline{a} = \underline{b}$ for simultaneous solution of Eq. (62) at the mesh points can be defined as for MWR where the jth element of $\underline{a}$ is the value of temperature, U_j, at the jth mesh point. The matrix $\underline{A}$ is tridiagonal with main diagonal elements $(2-h^2)$, with elements of the next diagonals above and below the main diagonal of -1, and with the other elements zero. Since the Dirichlet boundary conditions are homogeneous, the elements of $\underline{b}$ are $h^2 x_j$. Once the matrix problem $\underline{A}\ \underline{a} = \underline{b}$ has been solved for the value of temperature at the mesh points, $\underline{a}$, the finite difference solution is complete.

The systems of simultaneous equations $\underline{A}\ \underline{a} = \underline{b}$ for MWR and finite differences will be solved by Gaussian elimination. A subroutine, LEQT1F, was selected from the IMSL program library to perform the Gaussian elimination. LEQT1F performs Gaussian elimination with partial pivoting, equilibration, and the Crout algorithm (Ref 7:LEQT1F 1-4). LEQT1F also indicates when solution is not possible due to a singular matrix $\underline{A}$ and tests the solution $\underline{a}$ to insure it agrees with matrix $\underline{A}$ to a specified number of digits.

Transient Problem

The transient problem or initial value problem is the problem of a wall warming up. The problem stated in dimensionless form is for $0 \leq x \leq 1$

$$\frac{\partial^2}{\partial x^2} U(x,t) = \frac{\partial}{\partial t} U(x,t) \tag{63}$$

$$U(0,t) = 1 \quad U(1,t) = 0 \tag{64}$$

$$U(x,0) = 0 \tag{65}$$

where Eq. (63) is the partial differential equation, where Eq. (64) is the Dirichlet boundary conditions, where Eq. (65) is the initial condition, and where U(x,t) is the dimensionless temperature. The exact solution to the problem is

$$U(x,t) = 1 - x - \frac{2}{\pi} \sum_{n=1}^{\infty} \frac{\sin(n\pi x)}{n} e^{-n^2\pi^2 t} \tag{66}$$

where 1-x represents the steady state solution that the whole solution decays to as time increases and where the balance of Eq. (66) represents the transient solution (Ref 1:93-96).

The MWR solution will be of the reduction to ordinary differential equations form. The trial functions will be defined as in Eq. (27) where the space part of the trial function satisfies the boundary conditions and the undetermined function of time is found to satisfy the differential equation and initial conditions. The nonhomogeneous

boundary conditions can be satisfied as in Eq. (21) by a series of trial functions which is zero on both boundaries and a leading function which reduces to the value of the solution on both boundaries. Then we can choose a trial function as

$$U_N(x,t) = 1 - x + \sum_{i=1}^{N} A_i(t)(x^i - x^{i+1}) \qquad (67)$$

where $U_N(x,t)$ is the trial solution, where 1-x satisfies the nonhomogeneous boundary conditions, where the spacial trial functions, $(x^i - x^{i+1})$, satisfy the homogeneous boundary conditions, and where $A_i(t)$ is the undetermined function of time. The spacial trial function, $(x^i - x^{i+1})$, is the same one used for the steady state problem.

The first step in the transient MWR solution is forming the matrices for the matrix eigenvalue problem, $\underline{G}\,\underline{a} = \lambda\,\underline{H}\,\underline{a}$, where $\underline{G}$ is the coefficient matrix that corresponds to the left side of Eq. (63), $\underline{H}$ is the matrix for the right side of Eq. (63), $\underline{a}$ is the vector of $A_i(t)$ time functions, and where λ is the eigenvalue. First, form the residual of the differential equation by substituting the trial function Eq. (67) into Eq. (63):

$$0 + \sum_{i=1}^{N} A_i(t)[i(i-1)x^{i-2} - (i+1)ix^{i-1}]$$

$$= 0 + \sum_{i=1}^{N} [x^i - x^{i+1}] \frac{\partial A_i(t)}{\partial t} \qquad (68)$$

Note that the residual of the leading term of the trial function is zero. Thus, the problem becomes an eigenvalue problem as in Eq. (39). The form of the time function, $A_i(t)$, for a problem of this type is

$$A_i(t) = \sum_{k=1}^{N} C_k \, a_{ik} \, e^{\lambda_k t} \tag{69}$$

where a_{ik} represents one element of the kth eigenvector of $\underline{a}$, where C_k represents the multiplier of the kth eigenvector, and λ_k is the eigenvalue associated with the kth eigenvector. Since the leading term of the trial solution is the steady state solution, change the form of $A_i(t)$ to indicate it is decaying in time:

$$A_i(t) = \sum_{k=1}^{N} C_k \, a_{ik} \, e^{-\lambda_k t} \tag{70}$$

Then the residual Eq. (67) becomes

$$\sum_{i=1}^{N} [i(i-1)x^{i-2} - (i+1)ix^{i-1}] \, \underline{a}_k$$

$$= - \lambda_k \sum_{i=1}^{N} [x^i - x^{i+1}] \, \underline{a}_k \tag{71}$$

where $\underline{a}_k$ is the kth eigenvector of $\underline{a}$. Minimizing the residual with the MWR weight function provides the matrices to solve for λ_k and $\underline{a}_k$:

$$\sum_{i=1}^{N} \int_0^1 w_j [i(i-1)x^{i-2} - (i+1)ix^{i-1}] \, dx \, \underline{a}_k$$

$$= \lambda_k \sum_{i=1}^{N} \int_0^1 w_j [x^{i+1} - x^i] \, dx \, \underline{a}_k \tag{72}$$

where w_j is the weight function. The left side of Eq. (72) is matrix $\underline{G}$ and the right side $\underline{H}$ for the eigenvalue problem $\underline{G}\,\underline{a} = \lambda\,\underline{H}\,\underline{a}$.

The two minimizing weight criteria for the transient problem are collocation and Galerkin. The $\underline{G}$ and $\underline{H}$ matrices for collocation are

$$G(j,i) = \begin{cases} -2\,, & i=1 \\ [(i-1)ix_j^{i-2} - (i+1)ix_j^{i-1}], & i=2,\ldots N \end{cases} \tag{73}$$

$$H(j,i) = (x_j^{i+1} - x_j^i) \tag{74}$$

where $G(j,i)$ and $H(j,i)$ are elements of matrices $\underline{G}$ and $\underline{H}$ and where x_j is the jth of N equally spaced collocation points on the interval $0 < x < 1$. The Galerkin matrices are

$$G(j,i) = \frac{i(i-1)}{(j+i-1)} - \frac{2i^2}{(j+i)} + \frac{(i+1)i}{(j+i+1)} \tag{75}$$

$$H(j,i) = \frac{2}{(i+j+2)} - \frac{1}{(i+j+1)} - \frac{1}{(i+j+3)} \tag{76}$$

The Galerkin matrices are in general symmetric, but the collocation matrices are not. Least squares was not used since it has λ^2 terms.

After the matrices are formed the eigenvalue problem is solved by IMSL library subroutine EIGZF which solves the real eigenvalue problem $\underline{G}\,\underline{a} = \lambda\,\underline{H}\,\underline{a}$ for the real or

complex eigenvalues and eigenvectors (Ref 7:EIGZF 1-5). The routine reduces G to upper Hessenberg form and B to upper triangular form. Then the routine transforms G to quasi-upper triangular form (upper Hessenberg with no two consecutive subdiagonal elements being nonzero) while keeping H in upper triangular form. Last, the routine calculates eigenvalues and eigenvectors through an iterative operation (Ref 10:241-255). The IMSL routine includes a performance index to evaluate how well the problem was solved and indicates if the routine cannot converge to one of the eigenvalues (Ref 7:EIGZF 2-3).

The possibility of complex eigenvalues and eigenvectors requires a strategy for reducing all eigenvalues to real and the eigenvectors to real and linearly independent. The eigenvalues of real symmetric matrices are real so the Galerkin method should have all real eigenvalues and eigenvectors (Ref 15:24). The collocation method may have some complex eigenvalues and eigenvectors due to its nonsymmetric matrices. A useful approximation for complex eigenvalues is the real part (Ref 4:739-740). Since complex eigenvalues come in conjugate pairs with eigenvectors that are also conjugates, taking the real part of the eigenvalue results in a repeated real eigenvalue. Two linearly independent eigenvectors are obtained by taking the real part of the eigenvector for one and the imaginary part for the other (Ref 9:228-230).

After the eigenvalues and eigenvectors have been produced and made real and linearly independent, the initial conditions are used to find the appropriate constant multipliers for the eigenvectors. The multipliers are the C_k of the Eq. (70) form of the time function, $A_i(t)$. The procedure in the last two paragraphs has found the eigenvalues, λ_k, and the eigenvectors, a_{ik}. Now the initial condition of Eq. (65) is minimized with the same weight functions to provide simultaneous equations for C_k. Substituting Eqs. (70) and (67) into Eq. (65) and integrating with a weight function, w_j, yields

$$\int_0^1 w_j(1-x)dx + \int_0^1 w_j \left[\sum_{i=1}^{N} \sum_{k=1}^{N} (x^i - x^{i+1}) C_k \, a_{ik} \, e^{-\lambda_k(0)} \right] dx$$

$$= \int_0^1 w_j(0)dx \qquad (77)$$

where N different w_j determine N different equations, where the left side is the minimization of the trial solution of Eq. (67), and where the right side is the minimization of the initial value. Note time takes its initial value zero in Eq. (77). The residual equation, Eq. (77), can be rearranged to provide simultaneous equations for C_k by exchanging the order of integration and summation:

$$\sum_{k=1}^{N} C_k \left[\sum_{i=1}^{N} a_{ik} \int_0^1 w_j(x^i - x^{i+1})dx \right]$$

$$= -\int_0^1 w_j(1-x)dx \qquad (78)$$

Then the coefficients for C_k are obtained by summing the elements of the kth eigenvector multiplied by the integral on the left side of Eq. (78). Eq. (78) then becomes a matrix problem, $\underline{M}\ \underline{c} = \underline{m}$, where M is the coefficient matrix of the left side of Eq. (78), where $\underline{c}$ is the vector of C_k, and where $\underline{m}$ is the vector of Eq. (78) right sides. $\underline{M}\ \underline{c} = \underline{m}$ is solved by Gaussian elimination for the multipliers, C_k, and the solution is then complete.

The integrals of Eq. (78) are evaluated analytically to provide recursion relations for $\underline{M}$ and $\underline{m}$. For collocation the recursion relations are

$$M(j,i) = x_j^i - x_j^{i+1} \tag{79}$$

$$m(j) = x_j - 1 \tag{80}$$

where M(j,i) is the integral of the (j,i) element of $\underline{M}$, m(j) is the jth element of $\underline{m}$, and where x_j is the jth of N equally spaced collocation points. For Galerkin the recursion relations are

$$M(j,i) = \frac{1}{(j+i+1)} - \frac{2}{(j+i+2)} + \frac{1}{(j+i+3)} \tag{81}$$

$$m(j) = \frac{2}{(j+2)} - \frac{1}{(j+1)} - \frac{1}{(j+3)} \tag{82}$$

The finite difference solution for the transient problem will be constructed using an explicit four point method. The differential equation, Eq. (64), will be approximated with a first forward difference in time and a central difference in space:

$$\frac{U_{j,k+1} - U_{j,k}}{h_t} = \frac{U_{j+1,k} - 2U_{j,k} + U_{j-1,k}}{h_x^2} \qquad (83)$$

where j is the index for mesh point location on x with uniform spacing of h_x between mesh points and where k is the index for time mesh spacing of h_t between time levels (Ref 15:107-108). Solving for unknown temperature, $U_{j,k+1}$, gives the explicit equation

$$U_{j,k+1} = R(U_{j+1,k} + U_{j-1,k}) + (1 - 2R)U_{j,k} \qquad (84)$$

where $R = h_t/h_x^2$. Eq. (84) is used to solve for the temperature at a new time level repeatedly until the desired time is reached. For the explicit method to be stable, the time step size must obey the stability criteria $h_t \leq h_x^2/2$ (Ref 15:108). Since the difference expression error is $0(h_t) + 0(h_x^2)$, obeying the stability criteria should result in low error with neither space nor time error dominating the total error (Ref 15:108).

IV Results and Comparisons

The results of MWR and finite difference solutions to the two problems are compared with each other in this chapter. The results are also compared with the article by Houstis et.al. which compares MWR applied on finite elements with finite differences (Ref 6).

The Houstis et.al. article compares collocation, Galerkin, least squares, and finite differences for linear second order elliptic partial differential equations. The MWR methods used a rectangular grid to define the finite elements and Hermite bicubic polynomials for approximation of the solution. Solution of equations was by Gaussian elimination by profile or frontal method. The article conclusions state collocation was more efficient than finite differences for accuracy of one to four significant figures and beyond. The measure of efficiency was accuracy of solution and execution time. Finite differences and collocation started with equal efficiency at low accuracy or, in some cases, finite differences was most efficient at the lowest accuracy. As accuracy increased collocation became more efficient than finite differences after some crossover point at one to four significant figures of accuracy. Accuracy was obtained by measuring the error of the solution at the nodes of the mesh used for the finite element and finite difference statement of the problem (Ref 10:323-334).

The comparison of Galerkin and least squares methods to collocation shows that collocation is always faster for

equal accuracy with less of an advantage as finer grids are used. The reason for the speed advantage of collocation was attributed to its narrower band matrix than Galerkin or least squares. This advantage was true even though the symmetry properties of the Galerkin and least squares coefficient matrices were used to reduce solution time by half with the Cholesky decomposition for band matrices (profile method). For a given mesh size collocation was never more accurate than the other methods. The main conclusion of the article is that collocation is the best method of the four for the class of problems examined (Ref 6:335-337).

Measurement of Accuracy

The measure of accuracy used for comparisons in this study is the maximum absolute error. The maximum error encountered in all points tested becomes the error of that solution. Due to the uniform convergence properties of MWR, the maximum error is expected to decline with more terms in the trial solution as finite difference error declines with more mesh points (Ref 8:263-265,273). The fractional error has no such expectation. Fractional error is a measure of the number of significant figures in the MWR or finite difference solution. Significant figures are equal to -log (error/exact solution) where error/exact solution is fractional error (Ref 6:333). Since fractional error depends on the value of the exact solution as well as the error at a point, it will not be used to compare efficiency between methods.

Fractional error has been used by Houstis et.al. as a characteristic of solution comparisons. After finding the crossover point for finite difference and collocation a characteristic number of digits is assigned to describe the solutions at that point. At the crossover point, maximum error and execution time are approximately equal with collocation obtaining greater accuracy in less time after the crossover point. The number of digits assigned to the crossover point is -log(maximum error/maximum solution size). This number of digits is clearly an optimistic estimate of the number of significant figures in the solution since the fractional error of the maximum error is computed at the largest value of the exact solution instead of where the maximum error occurred (Ref 6:333).

The absolute error must be measured at a number of points in the domain to find the maximum error. Houstis et. al. measured the error at the nodes or intersections of the finite element grid (Ref 6:332). These are the mesh points of finite difference for a grid of the same size. Since no finite element mesh is used for the problems considered here, the finite difference points will be used for comparison. For collocation these points are the same as the collocation points. Then for an N term expansion of the MWR trial solution, the error will be measured at N equally spaced points, initially. A more extensive investigation of the MWR solutions will be made at 99 points throughout the domain of the problem to check the error obtained at the finite difference mesh points.

Steady State Problem

The solution to the steady state problem is shown in Figure 1. The exact solution is shown in comparison to the one term and two term collocation solutions. The convergence of the MWR solutions to the true solution is rapid with successive expansions approximating the true solution more closely.

In order to compare the maximum error of the MWR and finite difference solutions, 20 different solutions were computed for each MWR method and finite differences. Each MWR method used solution expansions ranging from one to 20 terms. The finite difference solutions ranged from one to 20 mesh points. The results of maximum error and execution time measurements are tabulated in Table I and Table III through Table V in Appendix A. The results are also presented graphically in Figures 2 and 3.

Figure 2 shows the maximum absolute error obtained by measurement at the finite difference mesh points for all four methods compared. The one term expansions of MWR were compared with the true solution at one point. The two term expansions were compared at two points and so on. The maximum error is plotted against execution time for 100 executions of each solution. Each symbol represents a different solution with a different number of terms or mesh points. The one term and one mesh point solutions are at the top of each line. The three MWR methods performed similarly. Galerkin was slightly faster than least squares

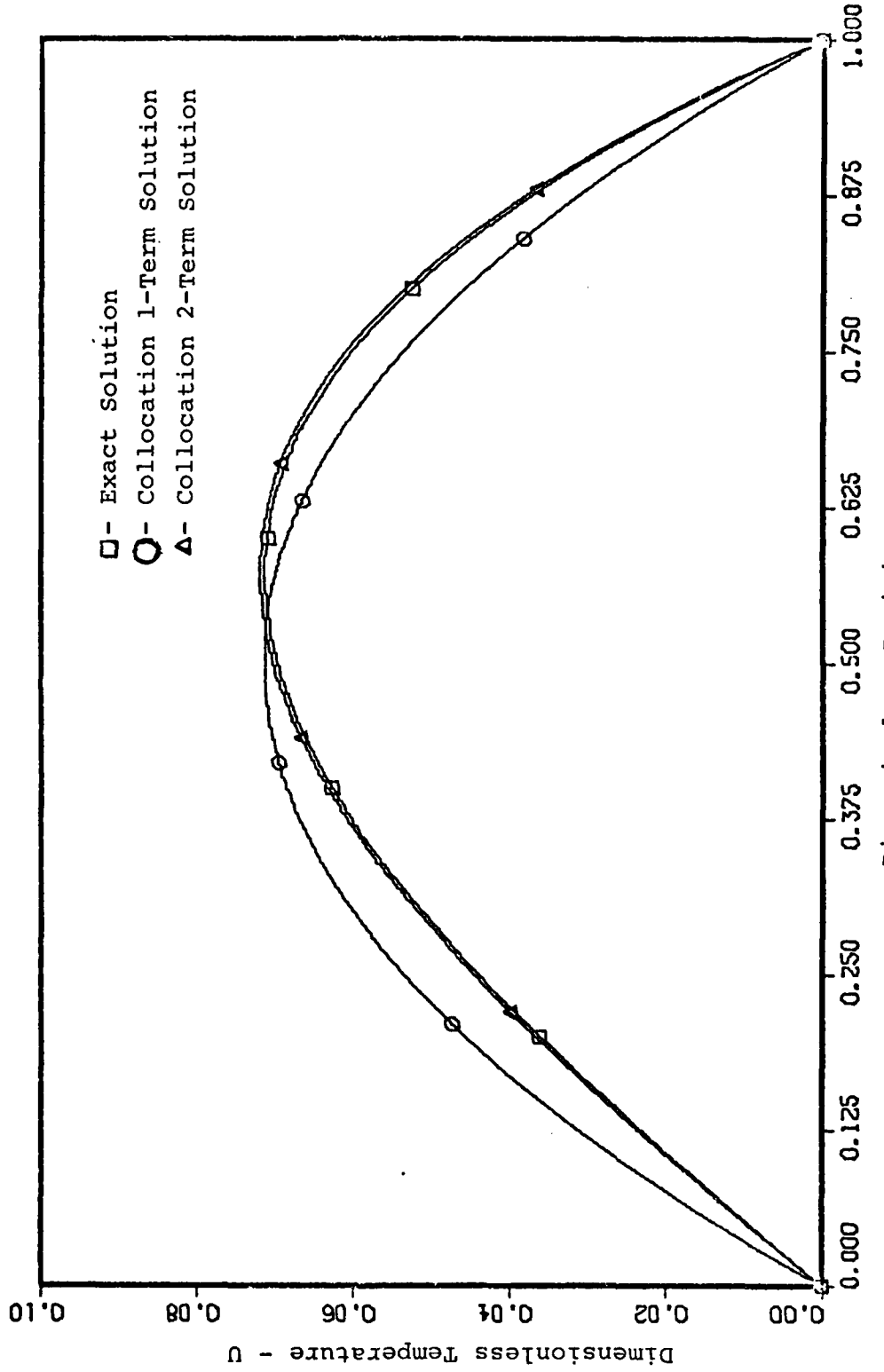

Fig 1. Steady State Problem Solution

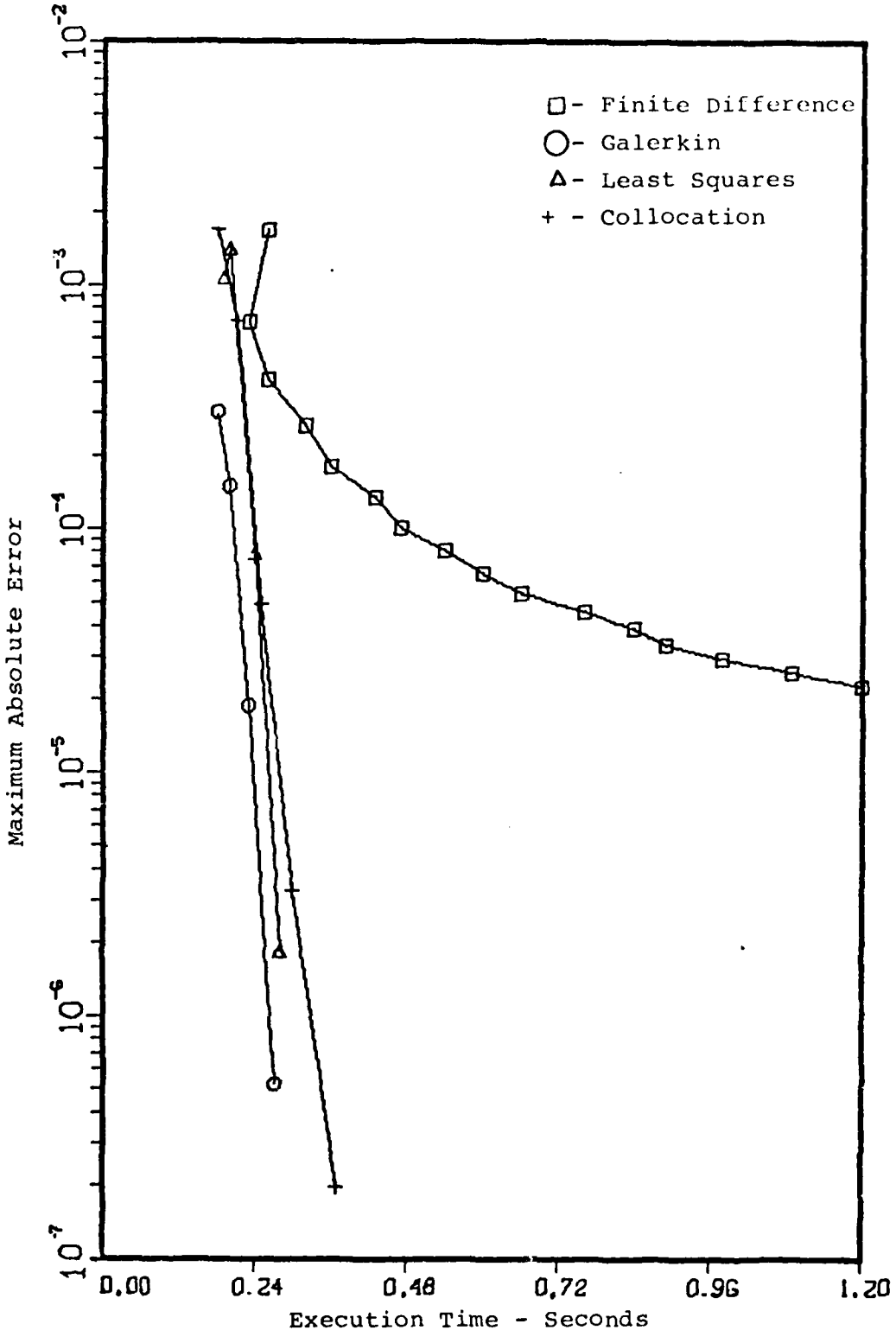

Fig 2. Steady State Error at Finite Difference Mesh Points

and collocation in the maximum error range around 10^{-3} to 10^{-5} with least squares approaching Galerkin below 10^{-5}. Collocation continues slower than Galerkin or least squares at smaller errors. The finite difference solutions shown in Figure 2 are much slower to a given maximum error than the MWR methods for errors less than $5x10^{-4}$. A characteristic number of digits accuracy can be assigned to the crossover point for this problem at between 10^{-3} and $7x10^{-4}$ maximum error. Since the maximum solution size for this problem is .071, the digits of accuracy for the crossover is between 1.8 and 2.0. This accuracy is a very optimistic estimate of the accuracy of the solution. For a point other than the crossover different accuracies require widely varying running times by different methods. At maximum error of $7x10^{-5}$ or 3.0 digit accuracy, finite differences takes twice as long as the MWR methods.

Figure 3 shows maximum error plotted against execution time for 100 executions when maximum error is determined by sampling the MWR solutions at 99 points. The absolute error of each solution was measured at the same 99 equally spaced points throughout the domain. The finite difference comparison at the mesh points of Figure 2 is included for reference. Although the maximum error measured for MWR solutions of a few terms is a few to several times higher for 99 points than for the finite difference points, the difference is small for solutions of 10 terms or more. The efficiency and relative efficiency of the MWR solutions is

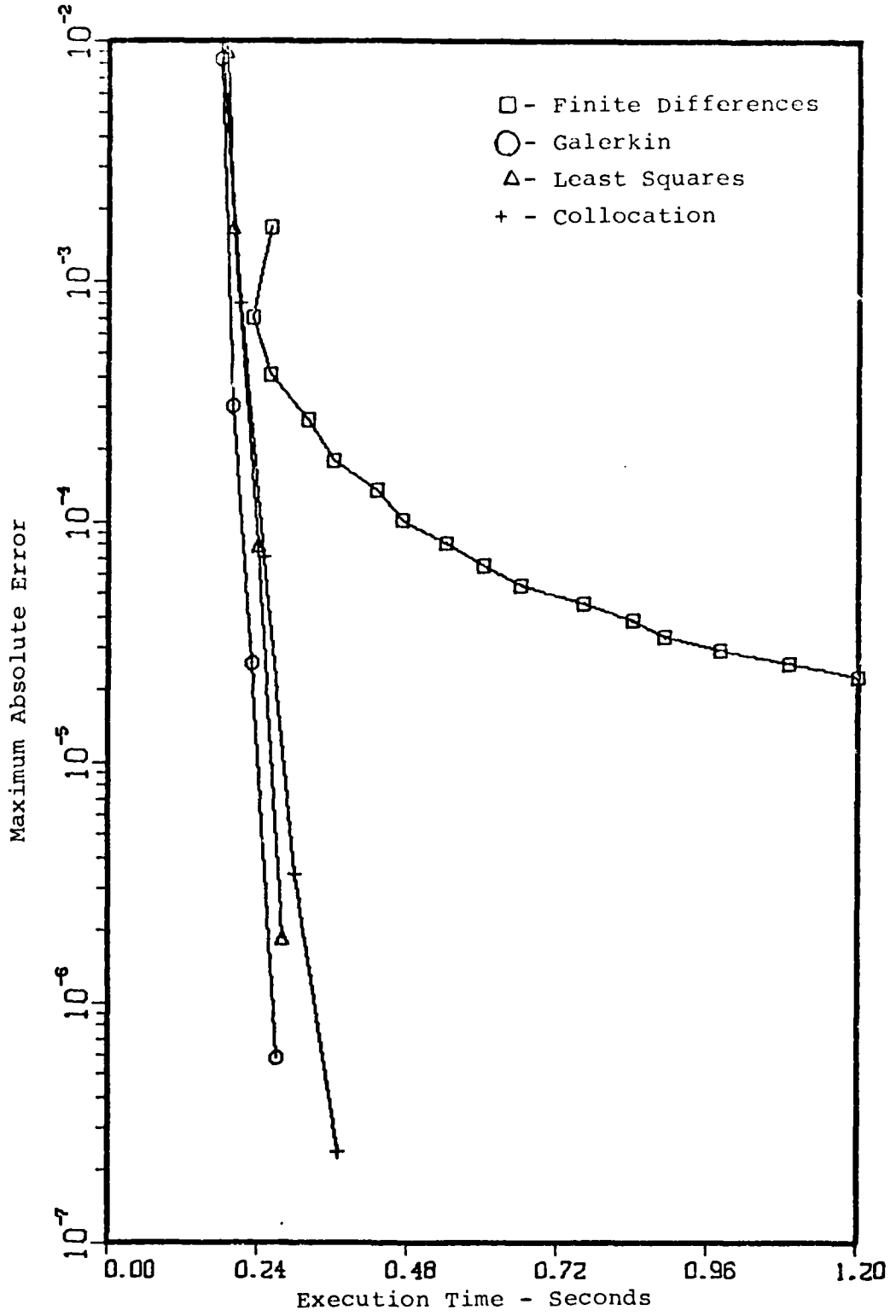

Fig 3. Steady State Error at 99 Points

almost the same for measurement at the 99 points as the finite difference points.

The Houstis et.al. article found collocation to be faster to a given accuracy than Galerkin or least squares for MWR applied on finite elements (Ref 6:335-337). In this formulation of MWR on the whole domain and for this problem, Galerkin is faster than either least squares or collocation. Also least squares is faster than collocation. The numerical integration used for Galerkin and least squares in MWR on finite elements and the resulting coefficient matrix may be the reasons for the different behavior. As indicated in the Houstis article, the numerical integration for MWR on finite elements requires evaluation of the integrand at nine places where collocation requires evaluation of the residual at only four points (Ref 6:336). The resulting equations produce coefficient matrices which take longer to solve for Galerkin and least squares than collocation (Ref 6:336). MWR on the whole domain as used in this study has neither of these two hindrances. The integrals are done analytically to develop recursion relations for matrix formation. The resulting matrices have all non-zero elements and are all solved by the same method so solution times are close to the same. Galerkin and least squares were observed to be more accurate for a given mesh size than collocation by Houstis et.al. (Ref 6:336). Apparently the greater accuracy of Galerkin and least squares results in less error for a given number of terms in the expansion than collocation.

Then the very close solution times result in smaller maximum error for Galerkin than for collocation.

Table I shows some of the values presented graphically in Figure 2 and Figure 3 for collocation. The maximum absolute error decreases with more terms in the trial solution until a minimum error is reached at 11 or 12 terms. After that point the error fluctuates and gradually grows. Galerkin and least squares exhibit similar behavior at eight or nine terms in the trial solution for minimum error. This behavior occurs well below the region of comparison of Figures 2 and 3. Galerkin and least squares error are tabulated in Table III and Table IV of Appendix A. All three solutions attained minimum errors of 10^{-12} or less.

There are three possible explanations for the behavior of maximum error. The first explanation of convergence to some solution other than the true solution is not valid. If the three methods converged to the wrong solution, maximum error would decrease to some minimum value and not grow. A second explanation is low accuracy of the Gaussian elimination solution of the simultaneous equations. A successive over-relaxation iterative method was used to improve the Gaussian elimination solution (Ref 15:126-129). A relative convergence test of 10^{-20} was used to test for convergence of the iterations. For collocation the successive over-relaxation gave greater error for some trial solutions and failed to yield a solution for others. For Galerkin and least squares, however, the method worked. Several solution constants changed value in the 10th to 11th significant figures.

TABLE I

Error and Time for Collocation on the Steady State Problem

Number of Terms in the Trial Solution	Maximum Absolute Error At Collocation Points	Maximum Absolute Error At 99 Points	Time for 100 Executions (seconds)
1	1.68E-3	9.69E-3	.18
2	7.09E-4	8.06E-4	.21
3	4.87E-5	7.06E-5	.25
4	3.23E-6	3.38E-6	.30
5	1.96E-7	2.38E-7	.37
6	7.73E-9	7.93E-9	.45
7	4.08E-10	4.65E-10	.53
8	1.16E-11	1.18E-11	.62
9	5.35E-13	5.90E-13	.71
10	1.46E-14	1.88E-14	.85
11	3.77E-15	6.77E-15	.98
12	5.77E-15	6.55E-15	1.11
13	8.44E-15	9.77E-15	1.25
14	4.22E-15	5.33E-15	1.42
15	6.22E-15	7.11E-15	1.62
16	6.22E-15	6.66E-15	1.81
17	4.85E-14	4.88E-14	2.01
18	2.29E-14	2.49E-14	2.23
19	5.77E-14	5.99E-14	2.45
20	1.38E-13	1.38E-13	2.65

Note: E-x means 10^{-x}.

The maximum errors of the solutions in Tables III and IV remained the same to three significant figures yielding Tables III and IV again. Since the growth in the error was up to two orders of magnitude, poor accuracy of the Gaussian elimination does not explain the behavior of maximum error. The last explanation of the maximum error behavior is the trial functions themselves. Babuska et.al. find these trial functions numerical unstable for the MWR in their study of stability in optimal trial functions (Ref 14:241-245). By studying the solution of the simultaneous equations based on these trial functions, they conclude they are unstable for seven or more terms in the trial solution. The solutions with more than seven terms have errors which increase, not decrease. Babuska et.al. conclude these trial functions should not be used for computer solutions.

Transient Problem

The transient problem was solved by the reduction to ordinary differential equations method of MWR and by an explicit finite difference method. The MWR on the whole domain produces a series solution that may be evaluated for any time or position in the domain. The explicit finite difference method, however, produces the solution for a succession of times up to the final time considered. For the comparison of the methods in this study, only the maximum absolute error at specified dimensionless times will be used.

Several points within the space domain were used to compare an approximate solution and the exact solution. The finite difference solutions were found for 10 to 100 mesh points in increments of 10. The finite difference mesh points were used to determine the maximum absolute error of that solution. The MWR solutions used successively more terms and solutions with 1 to 20 terms were calculated. The MWR solutions were compared at the collocation points for the steady state problem. Since most of the finite difference solutions in the transient problem were compared at more than the maximum number of collocation points, all the MWR solutions will be compared at 99 equally spaced points for a comparable search of the domain.

The solutions were compared at three times. Figure 4 shows the exact solutions at t = .05, .10, and .15. Note that the solution for t=.05 is close to zero for a larger fraction of the domain than the others and represents the earliest time response of the solution. The solution for t=.15 represents the latest time response when the solution is close to the steady state solution.

The maximum absolute error for the three solutions is plotted as a function of solution execution time in Figures 5 through 7. All solutions were executed 100 times for a more accurate measurement of the time. Only collocation and Galerkin MWR methods were us[illegible] Galerkin was always faster to a given accuracy than collocation as in the steady state problem. Both the Galerkin and collocation

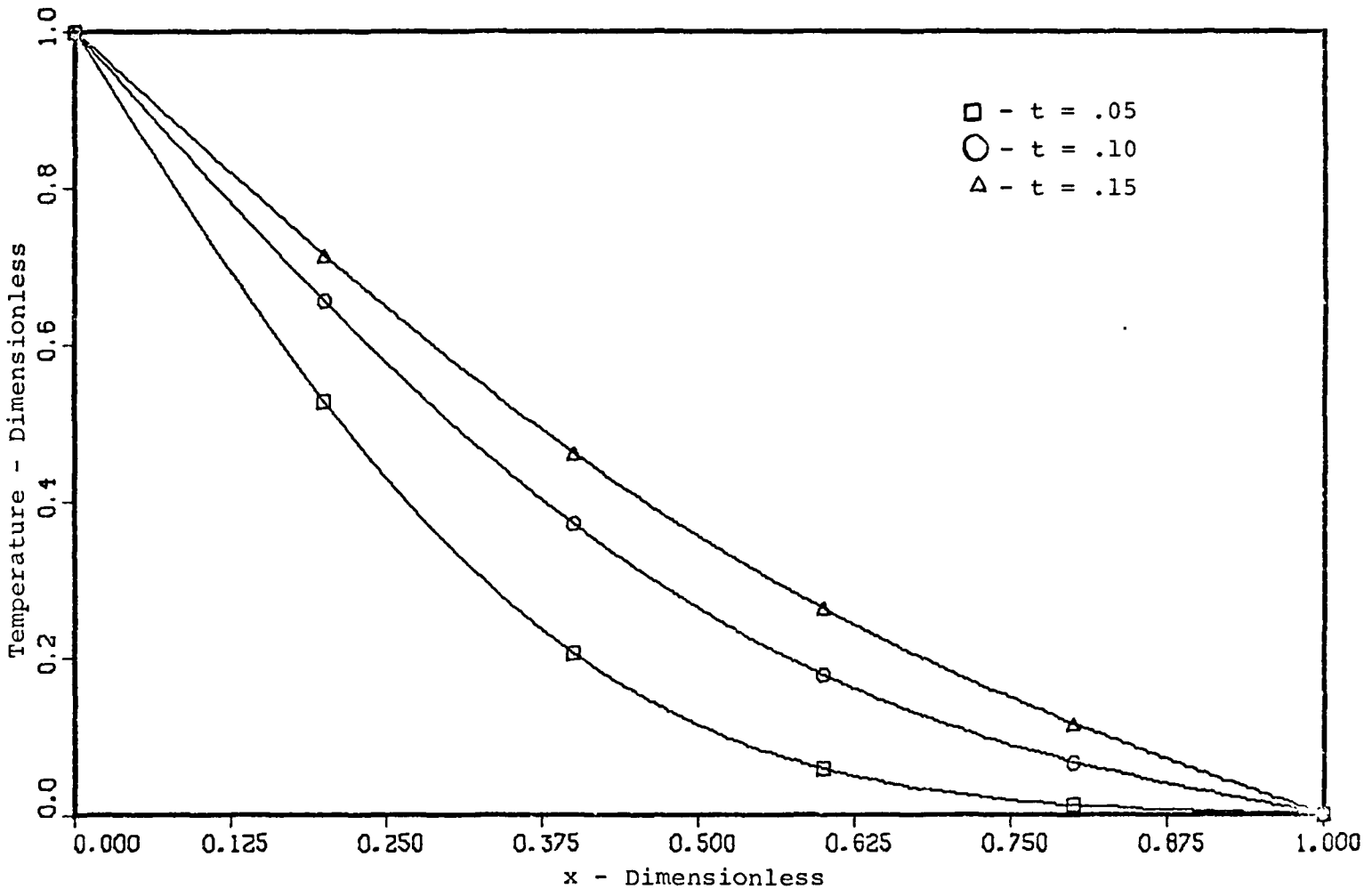

Fig 4. Transient Problem Exact Solution at Three Dimensionless Times

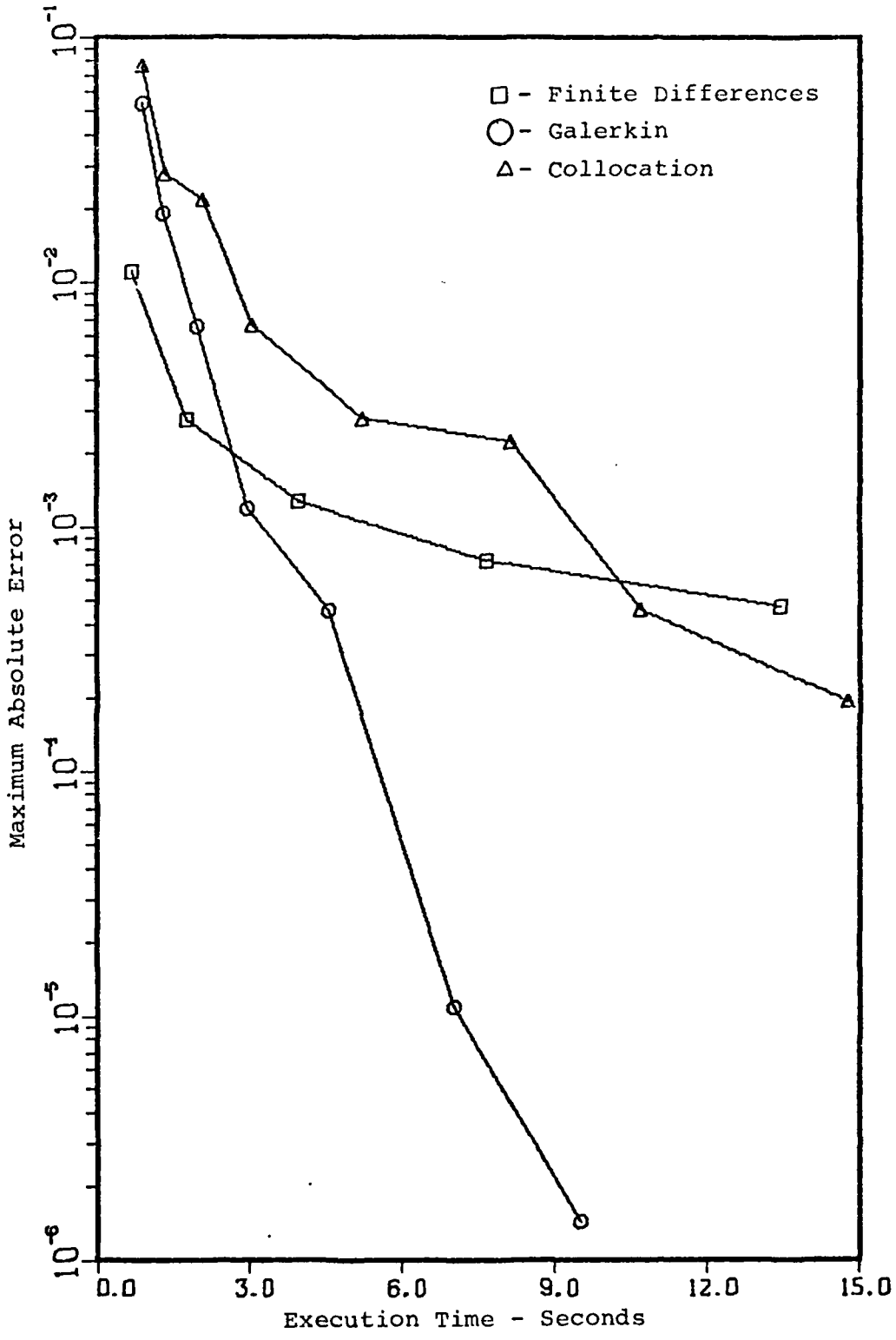

Fig 5. Transient Error for Time = .05

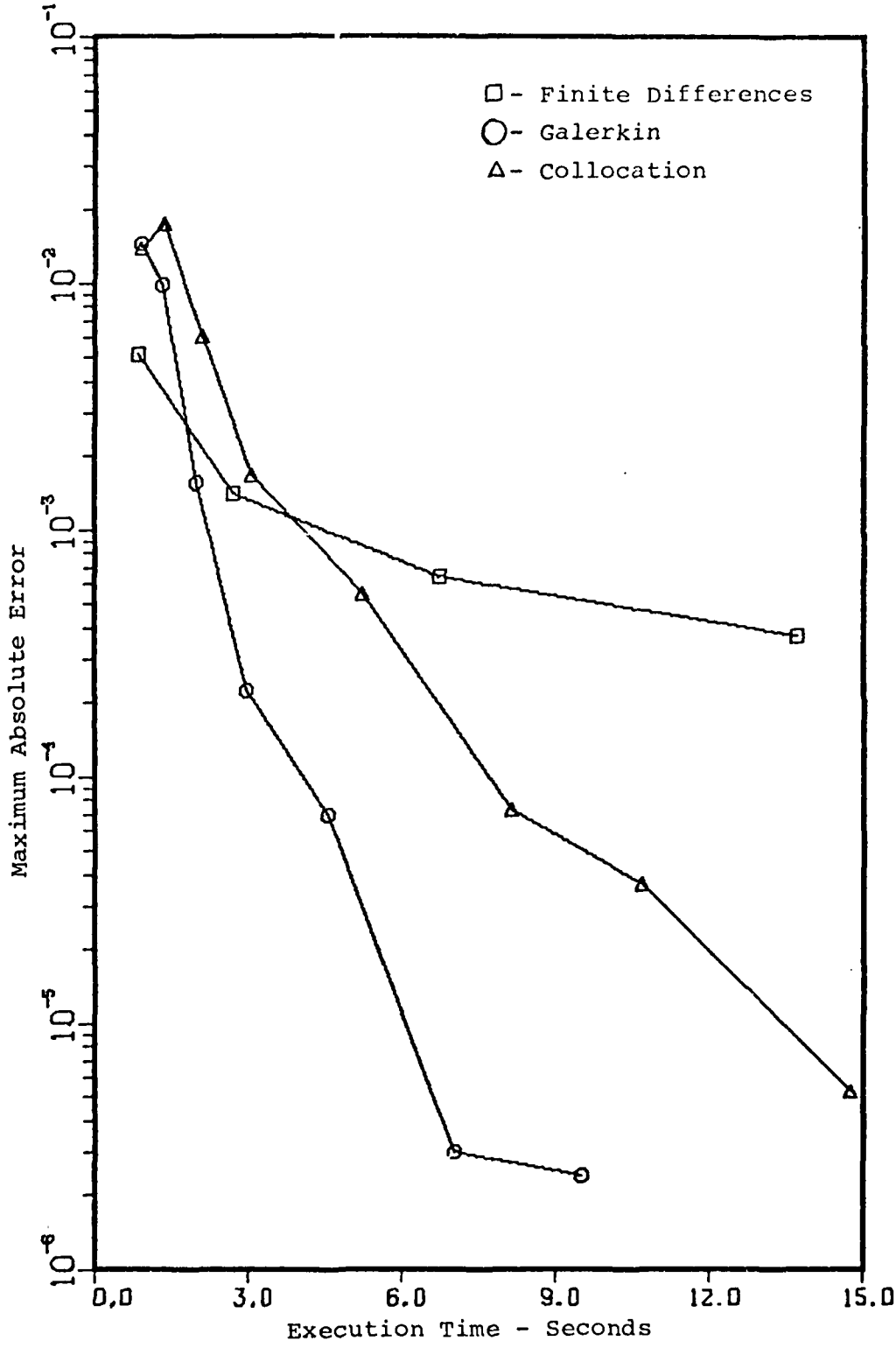

Fig 6. Transient Error for Time = .10

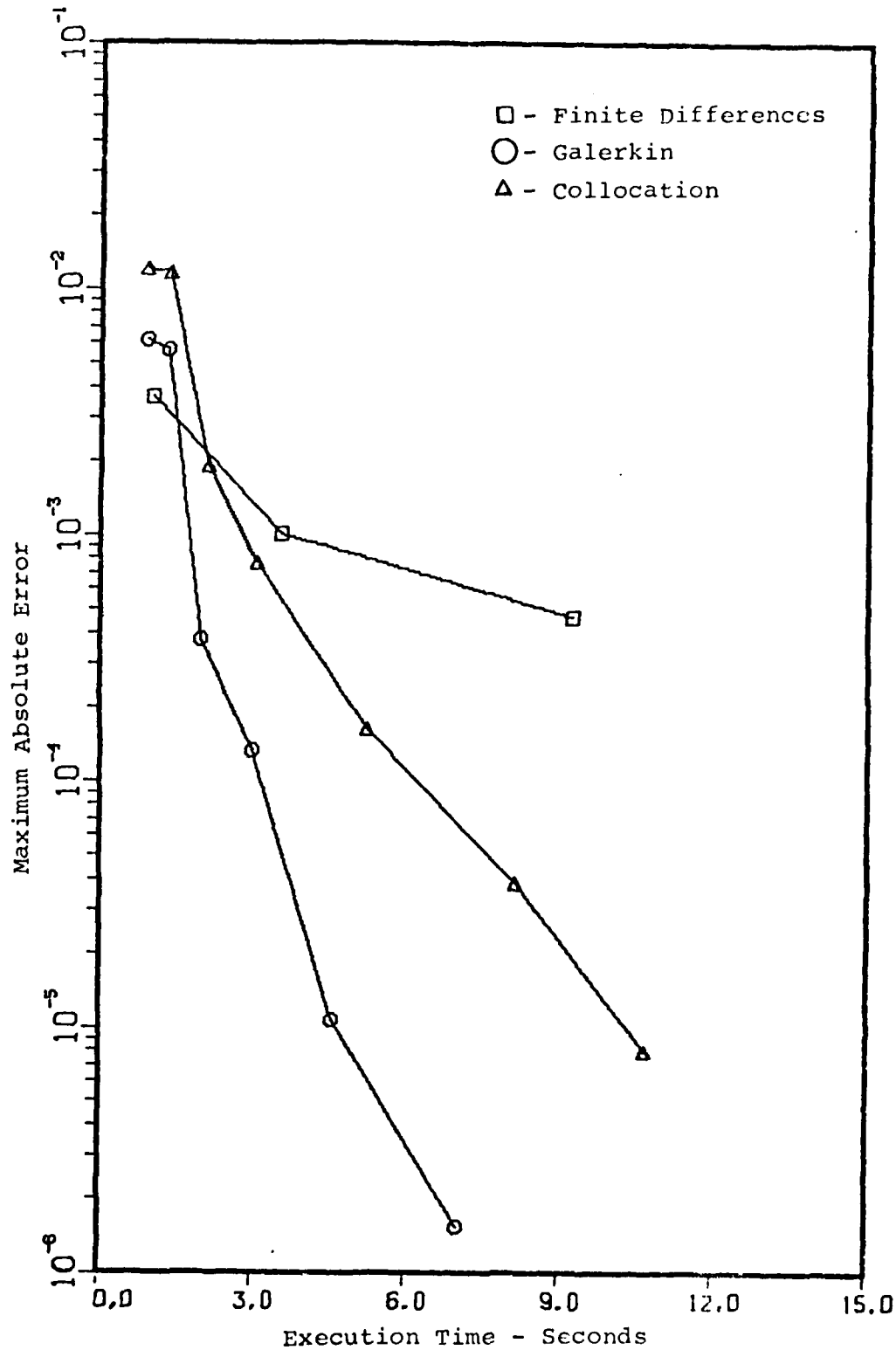

Fig 7. Transient Error for Time = .15

methods were faster to a given accuracy than explicit finite differences after a crossover point where MWR and finite differences achieved equal accuracy in equal time. The crossovers occurred at accuracies of 2.5 to 3.2 digits based on the maximum absolute error and maximum solution size of 1.0. The crossovers represent very optimistic estimates of accuracy. All methods showed improved accuracy for t=.10 and t=.15 over t=.05. Collocation improved the most and finite differences the least. The crossover point for collocation and finite differences changed from 3.2 digits at t=.05 to 2.6 digits at t=.15. The Galerkin and finite differences crossover remained more nearly constant by changing from 2.7 digits at t=.05 to 2.5 digits at t=.15. Collocation lost more of its advantage over finite differences at the short solution time t=.05 than did Galerkin.

An explanation for the better showing of Galerkin over collocation is the complex eigenvalues found for collocation. The Galerkin matrices in the eigenvalue determination part of the solution process were symmetric, while the collocation matrices were not. For collocation complex eigenvalues were first found for the five term expansion of the trial solution and expansions for more terms also had complex eigenvalues. Since only the real part of a complex eigenvalue was used as an approximation in the solution, some accuracy was lost. Accuracy was also lost for the Galerkin method due to negative real eigenvalues. These negative eigenvalues produced positive exponential functions

of time which grew instead of decayed with time. Since the negative eigenvalues produced functions which were not a physically real, decaying transient solution, those eigenvalues were set to zero. The negative eigenvalues first occurred for the 11 term Galerkin solution and for all solutions with more terms. Thus, the effect on accuracy of the negative eigenvalues for Galerkin occurred at finer approximations than the complex eigenvalues for collocation.

An explanation for the poorer showing of both MWR methods at the t=.05 time is the shape of the true solution at that time. The true solution there is close to zero for a greater fraction of the domain than the other two times. Since the true solution is close to zero for that longer fraction of the domain, it is harder to approximate by a polynomial. More terms and more solution time are required for a given accuracy or maximum absolute error.

Table II shows the maximum absolute error and execution times for collocation solutions at the three dimensionless times. Note that the maximum absolute error stops decreasing and starts growing for solutions of 13 or 14 terms. The error for Galerkin also grows starting at 13 terms in the solution as shown in Table VI of Appendix A. Since the same space trial functions were used for the solution expansions in the steady state and transient problems and since similar growth in error occurred in the steady state problem, this growth in error can be attributed at least partially to the numerically unstable trial functions

TABLE II

Error and Time for Collocation in the Transient Problem

Maximum Absolute Error

Number of Terms in the Trial Solution	T=.05	T=.10	T=.15	Time for 100 Executions (seconds)
1	7.69E-2	1.39E-2	1.21E-2	.90
2	2.76E-2	1.75E-2	1.15E-2	1.34
3	2.17E-2	6.15E-3	1.92E-3	2.11
4	6.66E-3	1.67E-3	7.68E-4	3.07
5	2.79E-3	5.55E-4	1.63E-4	5.22
6	2.24E-3	7.37E-5	3.89E-5	8.13
7	4.57E-4	3.69E-5	8.09E-6	10.69
8	1.95E-4	5.33E-6	8.71E-7	14.76
9	3.88E-5	1.71E-6	3.01E-7	19.46
10	1.51E-5	3.87E-7	1.88E-8	26.89
11	4.91E-6	8.98E-8	8.86E-9	34.20
12	1.24E-6	2.07E-8	7.07E-10	38.95
13	4.52E-7	4.13E-9	8.43E-10	47.26
14	7.82E-8	1.40E-8	8.83E-9	60.19
15	1.16E-7	6.38E-8	3.87E-8	72.17
16	5.35E-7	2.67E-7	1.60E-7	82.23
17	2.48E-6	1.53E-6	9.33E-7	---
18	9.07E-6	2.06E-6	8.51E-7	---
19	3.74E-5	2.36E-5	1.44E-5	---
20	2.35E-2	4.34E-3	8.00E-4	---

Note: E-x means 10^{-x}.

(Ref 14:241-245). No iterative improvement of the final solution was attempted, however.

An implicit finite difference method was used to see if the solution time could be shortened from the explicit method. Using a time step, $h_t = h_x^2$, where h_t is the time step size and where h_x is the space mesh spacing to minimize the finite difference error, $0(h_t) + 0(h_x^2)$ (Ref 14:108). The resulting execution times were several to 100 times greater than for the explicit solutions with slightly higher maximum absolute error. The largest increase in solution times occurred for the finest mesh spacing where the largest full matrices were solved for a new temperature more times than the smaller matrices for larger mesh sizes.

No direct comparison with the Houstis et.al. article is possible for the transient problem since that article treated only steady state problems on finite elements (Ref 6). It is interesting to note that the Galerkin method proved faster to a given accuracy than collocation for both problems in this study while collocation was fastest for the MWR on finite elements of the Houstis et.al. article.

V Conclusions

In this chapter conclusions of the investigation of MWR on the whole domain and finite differences for solution of the heat equation will be presented. Also, recommendations for the improvement of the MWR methods used in this study and application of finite element methods to heat problems of axial symmetry will be presented.

Two, one dimensional problems of heat transfer were solved by the MWR on the whole domain of interest and by finite differences. One problem represented steady state heat transfer and the other transient heat transfer. Both problems used Dirichlet boundary conditions where the value of temperature was specified on the boundary. MWR methods for the steady state problem were collocation, Galerkin, and least squares. The least squares method was not used for the transient problem since it complicated the solution process. The finite difference formulation was implicit for the steady state problem and explicit for the transient problem. The integrals of the Galerkin and least squares methods were evaluated analytically to obtain recursion relations for MWR solution. All solutions were compared with the true solution to obtain maximum absolute error.

Conclusions and Limitations

The Galerkin method was fastest to a given accuracy or maximum absolute error of the methods evaluated. The crossover point for Galerkin and finite differences where

both had equal accuracy in equal time occurred at 1.8 to 2.0 digits of accuracy for the steady state problem and 2.5 to 2.7 digits for the times evaluated in the transient problem. The Galerkin method had greater accuracy than finite differences in equal time after the crossover point. The digits of accuracy are an optimistic estimate of the number of significant figures in the approximate solution based on the maximum absolute error occurring in the whole domain and the maximum value of the true solution. The digits of accuracy indicate that finite differences is only faster for rather crude accuracy and Galerkin and the other MWR methods are faster for greater accuracy. The speed margin is sizeable. The time required to increase one digit of accuracy beyond the crossover point for the steady state problem is half as long for Galerkin as for finite differences. The Galerkin method is the best of methods evaluated to use for problems of this type when accuracy greater than two or three digits is required.

Some limitations to the conclusion of the previous paragraph must be noted. First only Dirichlet boundary conditions were used for the two problems evaluated. Other boundary conditions may be evaluated by finding trial functions which satisfy them or by minimizing the boundary residual along with the differential equation as shown in Eq. (10) of Chapter II. However, the conclusion of the last paragraph cannot be extended to other boundary conditions without some evaluation of problems with the other

boundary conditions. Another limitation is constant thermal conductivity. Constant thermal conductivity throughout the whole domain was assumed to reduce the heat equation to a dimensionless form. Problems involving varying thermal conductivity should be evaluated to see if Galerkin is the best method. A third limitation is the one dimensional cartesian coordinate geometry of both problems evaluated. Before assuming that Galerkin is always fastest for axial symmetry or two dimensional problems on the whole domain problems in those geometries should be evaluated. The last limitation is the method of integral evaluation used for the Galerkin and least squares methods. If a numerical rather than an analytical integration were used, the solution times and, therefore, the speed advantage of those two methods could change.

A major drawback of the Galerkin and other MWR methods on the whole domain in the problems evaluated is the polynomial trial functions. These trial functions were seen to be numerically unstable for solutions with 10 or more terms. The maximum absolute error increased not decreased for solutions with more than 10 terms or so. If heat generation terms complicated the shape of the temperature distribution, more terms could be needed in the MWR solution to approximate it. If more than 10 terms were needed, MWR might be limited to low accuracy and be slower than finite differences. A new approach is needed to avoid the drawback of the linearly independent but numerically unstable polynomial trial functions.

Recommendations

The orthogonal collocation method could solve the problem of numerical instability (Ref 3:357). It is recommended for steady state problems. Orthogonal collocation uses trial functions composed of orthogonal polynomials and collocation points defined by the roots of the highest order polynomial. The method can be made to fit other boundary condition than Dirichlet and is valid for planar, cylindrical or spherical geometry. The method can solve problems in terms of the values of the function approximated at the collocation points. This ability allows great flexibility as well as computational ease in problem solution. Orthogonal collocation has been shown to be as fast as the other MWR methods for a given accuracy (Ref 3:97,100).

As an example consider a one-dimensional cylindrical symmetry steady state heat transfer problem where temperature varies only with the radius:

$$\nabla^2 U(r) + g(r)U(r) = f(r) \qquad (85)$$

$$0 \leq r \leq 1$$

where U(r) represents dimensionless temperature and where g(r) and f(r) are functions defining heat generation. The problem can be formulated as symmetric in r:

$$\nabla^2 U(r) + g(r^2)U(r) = f(r^2) \qquad (86)$$

Then the trial solution for such a problem can be formulated for Dirichlet boundary conditions at r=1 as

$$U(r) = U(1) + (1-r^2) \sum_{i=1}^{N} a_i P_{i-1}(r^2) \qquad (87)$$

where a_i is an undetermined constant and $P_{i-1}(r^2)$ is an orthogonal polynomial. The $P_{i-1}(r^2)$ are defined starting with a power series in r^2 as

$$\int_0^1 w(r^2) P_j(r^2) P_i(r^2) r \, dr = C_i \delta_{ij} \qquad (88)$$

for j=1,2,...,i-1 where $w(r^2)$ is a weight function, δ_{ij} is the Kronecker delta function, and C_i some constant. Thus each polynomial in the trial function of Eq. (87) is made orthogonal to the others with the weight function, $w(r^2)$. The roots of the polynomials may be found for a given weight function. Weight function $(1-r^2)$ or 1 may be used for this problem. The polynomial roots for these weight functions have been tabulated (Ref 3:99,101-103).

For solution the trial function must be converted to an ordinary polynomial and matrices found to relate the values of temperature to the derivatives in the problem. Since $P_{N-1}(r^2)$ is a polynomial of degree N-1 in r^2, then the trial function can be represented as a polynomial of degree N in r^2:

$$U(r) = \sum_{i=1}^{N+1} d_i r^{2i-2} \qquad (89)$$

where d_i are new constants. Evaluating the derivative and Laplacian of temperature at the collocation points plus r=1 for the N+1 collocation point yields matrices to relate temperature and derivatives to the constants d_i:

$$U(r_j) = \sum_{i=1}^{N+1} x_j^{2i-2} d_i \tag{90}$$

$$\left.\frac{dU}{dr}\right|_{r_j} = \sum_{i=1}^{N+1} \left.\frac{dr^{2i-2}}{dr}\right|_{r_j} d_i \tag{91}$$

$$\left.\nabla^2 U\right|_{r_j} = \sum_{i=1}^{N+1} \left.\nabla^2 r^{2i-2}\right|_{r_j} d_i \tag{92}$$

In matrix form Eq. (90) is $\underline{U} = \underline{Q}\ \underline{d}$, Eq. (91) is $\underline{\frac{dU}{dr}} = \underline{C}\ \underline{d}$, and Eq. (92) is $\underline{\nabla^2 U} = \underline{D}\ \underline{d}$. The derivative and Laplacian can now be formulated in terms of the temperature at the collocation points:

$$\underline{\frac{dU}{dr}} = \underline{C}\ \underline{Q}^{-1}\ \underline{U} \tag{93}$$

$$\underline{\nabla^2 U} = \underline{D}\ \underline{Q}^{-1}\ \underline{U} \tag{94}$$

The expressions of Eq. (93) and (94) may be substituted directly into a differential equation such as Eq. (86) to produce a matrix problem for temperature at the collocation points. Then the temperature at the collocation points may be used to solve for the constants, d_i, in the trial solution. An additional condition needed for solution of Eq. (86) is the temperature at r=1 or $U(r_{NH}) = U(1)$. Other boundary conditions can be solved for $U(r_{N+1})$ and included in the solution for temperature (Ref 3:100-101).

Another method which should be evaluated in future research is the MWR on finite elements for problems of axial symmetry. MWR on finite elements has been shown to be

faster to a given accuracy than finite differences for steady state problems similar to the heat equation in two dimensions (Ref 6:336). Finite elements methods use piecewise continuous trial functions of a fixed number of terms. Accuracy is increased by increasing the number of elements covering the domain of the problem and not the number of terms in the trial function.

Problems of axial symmetry use volume finite elements defined in the two variables r and z. Axial symmetry is described by a cylindrical coordinate system where temperature or boundaries do not vary with the angular coordinate of the cylindrical coordinate system. The domain of the problem is then divided into finite element rings with cross section and trial functions described by r and z coordinates of the cylindrical coordinate system. For integrations involved in the MWR the volume of the finite element ring must be used. Various shape elements may be used with the appropriate trial functions to element shape and order of the polynomial trial function used. The trial solution is expressed as in orthogonal collocation as a function of parameters at specific nodes or points on the element:

$$U = \Sigma N_i a_i \tag{95}$$

where U is the function approximated such as temperature, N_i are the trial functions called shape functions in finite elements dependent on the shape and order of polynomial

approximation, and where a_i are the nodes determined by element shape and shape functions (Ref 13:119-120,148-177).

The MWR statement of the steady state heat transfer problem minimizes the combined residual of the differential equation and boundary conditions of second and third kind to produce a matrix statement for solution of the nodal parameters, a_i. When the matrix equations are solved the solution is complete. The matrix equations are of the form $\underline{H}\ \underline{a} + \underline{f} = 0$ where $\underline{H}$ is obtained by numerical integration of the integrals involving $\underline{a}$ and where $\underline{f}$ is obtained from integrals not involving $\underline{a}$ (Ref 13:424-426). The method of combining contributions to the $\underline{H}$ matrix from each finite element of the domain results in a banded, sparse matrix which is faster to solve than the full matrices of MWR on the whole domain (Ref 13:14-15).

The transient heat transfer equation may be treated in the same manner by MWR to produce ordinary differential equations in time. The matrix formation is then of the form $\underline{C}\ \dot{\underline{a}} + \underline{H}\ \underline{a} + \underline{f} = 0$ which may be solved by methods similar to those used for the transient problem in this study. A step by step recurrence calculation for a_i similar to finite difference methods for the transient problem is also possible and more general (Ref 13:569). In the step by step method shape functions are chosen to describe the variation of $\underline{a}$ from the beginning to the end of a time element, $\underline{a} = \sum_{i=n}^{n+1} N_i\ \underline{a}_i$. The MWR is applied to the whole matrix formulation of the problem with the time derivatives acting

on the shape functions. The resulting equation is then reduced to yield a recurrence relationship for $\underline{a}_{n+1}$ as a function of $\underline{a}_n$. Thus the solution can be produced in a step by step basis as for finite differences (Ref 13:570-572). Evaluation of the finite element methods described here should provide a better and more general test of MWR for heat transfer problems of axial symmetry.

Bibliography

1. Carslaw, H. S. and H. C. Jaeger. Conduction of Heat in Solids. Second Edition. Oxford University Press. London, England. 1959.

2. Crandall, S. H. Engineering Analysis. McGraw-Hill Book Co., Inc. New York. 1956.

3. Finlayson, B. A. The Method of Weighted Residuals and Variational Principles. Academic Press. New York. 1972.

4. Finlayson, B. A. and L. E. Scriven. "The Method of Weighted Residuals - A Review," Applied Mechanics Reviews, 19: 735-743 (September 1966).

5. Fortran Extended Version 4 Reference Manual Number 60497800. Control Data Corporation. Sunnyvale, California. 1979.

6. Houstis, E. N., R. E. Lynch, and J. R. Rice. "Evaluation of Numerical Methods for Elliptical Partial Differential Equations," Journal of Computational Physics, 27: 323-350 (1978).

7. IMSL Library Reference Manual, Edition 8. IMSL, Inc. Houston, Texas. 1980.

8. Kantorovich, L. V. and V. I. Krylov. Approximate Methods of Higher Analysis. Fourth Edition Translated from Russian by C. D. Benster. P. Noordhoff Ltd. Groningren, Netherlands. 1958.

9. Kaplan, W. Ordinary Differential Equations. Addison-Wesley Publishing Company, Inc. Reading, Massachusetts. 1958.

10. Moler, C. B. and G. W. Stewart. "An Algorithm for Generalized Matrix Eigenvalue Problems," SIAM Journal of Numerical Analysis, 10: 241-256 (1973).

11. Myers, G. E. Analytical Methods in Conduction Heat Transfer. McGraw-Hill Book Company. New York. 1971.

12. Ozisik, M. N. Boundary Value Problems of Heat Conduction. International Textbook Company. Scranton, Pennsylvania. 1968.

13. Zienkiewicz, O. C. The Finite Element Method. Third Edition. McGraw-Hill Book Company, Ltd. London, England. 1977.

14. Babuska, I. M., M. Prager, and E. Vitsek. Numerical Processes in Differential Equations. John Wiley and Sons, Inc. London, England. 1966.

15. Clarke, M., Jr. and K. F. Hansen. Numerical Methods of Reactor Analysis. Academic Press. New York. 1964.

16. Sparrow, E. M. and A. Haji-Sheikh. "Transient and Steady Heat Conduction in Arbitrary Bodies with Arbitrary Boundary and Initial Conditions," Journal of Heat Transfer, 90: 103-108 (February 1968).

Appendix A: Error and Solution Time Tables

This appendix contains tables summarizing the maximum absolute error and solution time for MWR and finite difference solutions of the steady state and transient heat transport problems.

TABLE III

Error and Time for Galerkin on the Steady State Problem

Number of Terms in the Trial Solution	Maximum Absolute Error: At Finite Difference Mesh Points	Maximum Absolute Error: At 99 Points	Time for 100 Executions (seconds)
1	3.03E-4	8.35E-3	.18
2	1.49E-4	3.04E-4	.20
3	1.87E-5	2.57E-5	.23
4	5.22E-7	5.88E-7	.27
5	3.30E-8	3.74E-8	.30
6	6.19E-10	6.40E-10	.36
7	2.89E-11	3.21E-11	.42
8	8.10E-13	8.14E-13	.46
9	5.43E-12	5.52E-12	.52
10	2.29E-11	2.50E-11	.58
11	3.80E-11	3.80E-11	.67
12	4.60E-11	4.75E-11	.75
13	3.72E-11	3.78E-11	.81
14	1.71E-10	1.74E-10	.91
15	1.69E-10	1.81E-10	1.00
16	6.45E-11	6.68E-11	1.12
17	6.13E-11	6.34E-11	1.25
18	1.36E-10	1.41E-10	1.37
19	9.86E-11	1.09E-10	1.47
20	1.03E-10	1.17E-10	1.62

Note: E-x means 10^{-x}.

TABLE IV

Error and Time for Least Squares on the Steady State Problem

Number of Terms in the Trial Solution	Maximum Absolute Error: At Finite Difference Mesh Points	Maximum Absolute Error: At 99 Points	Time for 100 Executions (seconds)
1	1.68E-3	9.06E-3	.19
2	1.42E-3	1.68E-3	.20
3	7.80E-5	7.90E-5	.24
4	1.33E-6	1.85E-6	.28
5	9.60E-8	9.87E-8	.35
6	1.33E-9	1.68E-9	.39
7	7.53E-11	7.84E-11	.49
8	9.94E-13	1.06E-12	.55
9	4.56E-13	4.84E-13	.65
10	8.58E-13	8.77E-13	.75
11	3.96E-12	3.98E-12	.86
12	2.71E-11	2.73E-11	.97
13	1.01E-11	1.03E-11	1.09
14	2.76E-11	3.14E-11	1.23
15	1.29E-11	1.31E-11	1.35
16	3.94E-11	4.12E-11	1.53
17	7.41E-11	7.48E-11	1.71
18	2.09E-10	2.40E-10	1.84
19	6.44E-11	6.73E-11	2.05
20	3.65E-11	3.77E-11	2.22

Note: E-x means 10^{-x}.

TABLE V

Error and Time for Finite Difference Method on the Steady State Problem

Number of Terms in the Trial Solution	Maximum Absolute Error At Finite Difference Mesh Points	Time for 100 Executions (seconds)
1	1.68E-3	.26
2	7.09E-4	.23
3	4.09E-4	.26
4	2.64E-4	.32
5	1.81E-4	.36
6	1.35E-4	.43
7	1.02E-4	.47
8	8.17E-5	.54
9	6.58E-5	.60
10	5.46E-5	.66
11	4.58E-5	.76
12	3.90E-5	.84
13	3.37E-5	.89
14	2.92E-5	.98
15	2.58E-5	1.09
16	2.28E-5	1.20
17	2.04E-5	1.32
18	1.83E-5	1.41
19	1.65E-5	1.53
20	1.50E-5	1.67

Note: E-x means 10^{-x}.

TABLE VI

Error and Time for Galerkin in the Transient Problem

Maximum Absolute Error

Number of Terms in the Trial Solution	T=.05	T=.10	T=.15	Time for 100 Executions (seconds)
1	5.39E-2	1.45E-2	6.18E-3	.91
2	1.92E-2	9.86E-3	5.65E-3	1.33
3	6.60E-3	1.56E-3	3.76E-4	1.99
4	1.19E-3	2.25E-4	1.34E-4	2.96
5	4.52E-4	6.98E-5	1.08E-5	4.57
6	1.09E-5	3.00E-6	1.53E-6	7.03
7	1.43E-6	2.41E-6	3.38E-7	9.51
8	7.01E-7	9.92E-8	1.32E-8	12.18
9	4.82E-7	5.53E-8	7.65E-9	16.01
10	3.84E-7	6.81E-9	2.24E-9	20.32
11	5.76E-8	5.45E-9	1.98E-9	24.80
12	1.70E-7	2.07E-9	4.44E-10	31.10
13	3.95E-8	1.35E-9	2.21E-10	38.47
14	8.13E-7	4.04E-8	2.80E-9	42.69
15	1.90E-7	1.26E-8	9.78E-10	50.99
16	4.18E-7	2.66E-7	1.68E-9	59.39
17	6.27E-7	1.71E-7	4.20E-8	68.25
18	5.75E-7	1.41E-7	4.62E-8	77.08
19	5.94E-7	1.41E-7	3.00E-8	---
20	1.11E-5	5.83E-6	3.09E-6	---

Note: E-x means 10^{-x}.

TABLE VII

Error and Time for the Explicit Finite Difference Method in the Transient Problem
T=.05

Number of Mesh Points	Maximum Absolute Error	Time for 100 Executions (seconds)
10	1.10E-2	.69
20	2.74E-3	1.77
30	1.28E-3	3.96
40	7.30E-4	7.66
50	4.72E-4	13.45
60	3.31E-4	21.59
70	2.44E-4	32.81
80	1.88E-4	47.00
90	1.49E-4	65.45
100	1.21E-4	88.21

Note: E-x means 10^{-x}. T is the dimensionless time variable.

TABLE VIII

Error and Time for the Explicit Finite Difference Method in the Transient Problem
T=.10

Number of Mesh Points	Maximum Absolute Error	Time for 100 Executions (seconds)
10	5.13E-3	.85
20	1.41E-3	2.69
30	6.53E-4	6.73
40	3.73E-4	13.71
50	2.41E-4	24.62
60	1.69E-4	40.46
70	1.25E-4	62.44
80	9.58E-5	91.17
90	7.59E-5	127.52
100	6.16E-5	---

Note: E-x means 10^{-x}. T is the dimensionless time variable.

TABLE IX

Error and Time for the Explicit Finite Difference Method in the Transient Problem
T=.15

Number of Mesh Points	Maximum Absolute Error	Time for 100 Executions (seconds)
10	3.65E-3	1.02
20	1.02E-3	3.55
30	4.69E-4	9.25
40	2.68E-4	19.57
50	1.73E-4	35.56
60	1.21E-4	58.99
70	8.96E-5	91.43
80	6.88E-5	135.13
90	5.45E-5	---
100	4.43E-5	---

Note: E-x means 10^{-x}. T is the dimensionless time variable.

Appendix B: Computer Programs

This appendix contains the computer programs for the steady state and transient heat transfer problems. Both the MWR and finite difference solutions are included.

```
      PROGRAM EASY1(INPUT,OUTPUT)
      DIMENSION  COEF(20,20),RS(20),WKAREA(20),FD(20)
      IDIM=20
      DO 1 ICRIT=1,3
      DO 1 N=1,20
      N1=N+1
      CALL MWRSM(COEF,RS,WKAREA,N,IDIM,ICRIT)
      CALL COMPMW(RS,N,IDIM)
      CALL FDSM(COEF,FD,WKAREA,N1,IDIM)
      CALL COMPFD(FD,N1,IDIM)
1     CONTINUE
      END
      SUBROUTINE MWRSM(COEF,RS,WKAREA,N,IDIM,ICRIT)
      DIMENSION COEF(IDIM,IDIM),WKAREA(IDIM),RS(IDIM)
      DIMENSION A(20,20),XI(20),Y(20)
      PRINT*,"     "
      PRINT*,"     "
      PRINT*,"     "
      PRINT*," MWR SOLN FOR ",N," TERM EXPANSION "
C   PICK METHOD
      IF (ICRIT.EQ.1) GO TO 1
      GO TO 20
1     PRINT*," COLLOCATION "
      CALL COL(COEF,RS,N,IDIM)
20    IF (ICRIT.EQ.2) GO TO 2
      GO TO 30
2     PRINT*," GALLERKIN "
      CALL GAL(COEF,RS,N,IDIM)
30    IF (ICRIT.EQ.3) GO TO 3
      GO TO 40
3     PRINT*," LEAST SQUARES "
      CALL LS(COEF,RS,N,IDIM)
C   PRINT RESULT OF CALL
40    PRINT*," RIGHT SIDE VECTOR= "
      PRINT*," COEF MATRIX= "
      DO 5 J=1,N
      DO 100 KK=1,N
100   A(J,KK)= COEF(J,KK)
      Y(J)= RS(J)
5     CONTINUE
C   FIND SOLN CONSTANTS
      ML=1
      IDGT= 8
      CALL LEQT1F(COEF,ML,N,IDIM,RS,IDGT,WKAREA,IER)
      PRINT*," IER= ",IER
      IF (IER.NE.34) GO TO 50
      PRINT*," EXACT METHOD SOLN CONSTANTS= "
      PRINT*,(" A(",J,")= ",RS(J),J=1,N)
      SORCON= 1.0E-8
      DO 60 J=1,N
60    XI(J)= RS(J)
      ..
```

```
      ALP=1.0
      CALL SOR(A,XI,RS,Y,WKAREA,N,IDIM,ALP,SORCON)
50    CONTINUE
      PRINT*,"      "
      PRINT*," SOLN CONSTANTS "
      PRINT*,(" A(",J,")= ",RS(J),J=1,N)
      PRINT*,"      "
      RETURN
      END
      SUBROUTINE COL(COEF,RS,N,IDIM)
      DIMENSION COEF(IDIM,IDIM),RS(IDIM)
C     FILL RS VECTOR
      DX= 1.0/(N+1)
      DO 1 J=1,N
1     RS(J)= DX*J
C     FILL COEF MATRIX
      DO 2 J=1,N
      DO 2 I=1,N
      X= DX*J
      COEF(J,I)= (I*(I+1)*(X**(I-1)))+(X**(I+1))-(X**I)
     1 - (I*(I-1)*(X**(I-2)))
2     CONTINUE
      RETURN
      END
      SUBROUTINE GAL(COEF,RS,N,IDIM)
      DIMENSION COEF(IDIM,IDIM),RS(IDIM)
C     FILL RS VECTOR
      DO 1 J=1,N
1     RS(J)= 1.0/((J+2)*(J+3))
C     FILL COEF MATRIX
      DO 2 J=1,N
      DO 2 I=1,N
      COEF(J,I)= (1.0*(((I+1)*I)+(I*(I-1)))/(J+I)) + (2.0/(J+I+2))
     1 - (1.0/(J+I+3))-(1.0*I*(I-1)/(J+I-1))-(1.0*((I*(I+1))+1)/(J+I+1))
2     CONTINUE
      END
      SUBROUTINE LS(COEF,RS,N,IDIM)
      DIMENSION COEF(IDIM,IDIM),RS(IDIM)
C     FILL RS VECTOR
      DO 1 J=1,N
1     RS(J)= 1.0 - (1.0/((J+2)*(J+3)))
C     FILL COEF MATRIX
      DO 2 J=1,N
      DO 2 I=1,N
      IF ((I+J).EQ.3) GO TO 11
      GO TO 20
11    COEF(J,I)=   (1.0*((J*(J-1))+(J*(J+1)*I*(I+1))+(I*(I-1)))/(I+J-1))
     1 - (1.0*((J*(J-1))+(J*(J+1))+(I*(I+1))+(I*(I-1)))/(I+J))
     2 + (1.0*((I*(I+1))+1.0+(J*(J+1)))/(J+I+1)) + (1.0/(I+J+3))
     3 - (2.0/(I+J+2))
     4  - (1.0*((J*(J-1)*I*(I+1))+(I*(I-1)*J*(J+1)))/(I+J-2))
      GO TO 90
20    IF ((I+J).EQ.2) GO TO 21
      GO TO 30
..
```

```
21    COEF(J,I)=  (1.0*(J*(J-1)*I*(I-1))/(I+J-3)) - (2.0/(I+J+2))
     2  + (1.0*((J*(J-1))+(J*(J+1)*I*(I+1))+(I*(I-1)))/(I+J-1))
     3  - (1.0*((J*(J-1))+(J*(J+1))+(I*(I+1))+(I*(I-1)))/(I+J))
     4  + (1.0*((I*(I+1))+1.0+(J*(J+1)))/(J+I+1)) + (1.0/(I+J+3))
      GO TO 90
   30 COEF(J,I)= (1.0*(J*(J-1)*I*(I-1))/(I+J-3)) - (2.0/(I+J+2))
     1  - (1.0*((J*(J-1)*I*(I+1))+(I*(I-1)*J*(J+1)))/(I+J-2))
     2  + (1.0*((J*(J-1))+(J*(J+1)*I*(I+1))+(I*(I-1)))/(I+J-1))
     3  - (1.0*((J*(J-1))+(J*(J+1))+(I*(I+1))+(I*(I-1)))/(I+J))
     4  + (1.0*((I*(I+1))+1.0+(J*(J+1)))/(J+I+1)) + (1.0/(I+J+3))
90    CONTINUE
2     CONTINUE
      RETURN
      END
      SUBROUTINE COMPMW(RS,N,IDIM)
      DIMENSION RS(IDIM)
C   COMPARE AT MESH POINTS/COLLOCATION POINTS
      PRINT*," ERROR OF MWR AT MESH POINTS OF FD OR COLLOCATION POINTS"
      ITOL=0
      M=N
      DX=1.0/(N+1)
      GO TO 50
30    IF (ITOL.EQ.1) GO TO 31
      GO TO 40
31    M=99
      DX= .01
      PRINT*," ERRORS FOR WHOLE REGION AT 99 POINTS "
      GO TO 50
40    IF (ITOL.EQ.2) RETURN
   50 ERRM= 0.0
      FERRM= 0.0
      XME=0.0
      XMF= 0.0
      DO 1 J=1,M
      X= DX*J
      Y= (SIN(X)/SIN(1.0)) - X
      YM= 0.0
      DO 10 I=1,N
10    YM= YM + (RS(I)*((X**I)-(X**(I+1))))
      ERR=Y-YM
      FERR=ERR/Y
      IF (ABS(ERR).GT.ERRM) GO TO 2
      GO TO 5
2     ERRM= ABS(ERR)
      XME= X
5     IF (ABS(FERR).GT.FERRM) GO TO 3
      GO TO 6
3     FERRM= ABS(FERR)
      XMF=X
6     IF (ITOL.EQ.2) GO TO 1
1     CONTINUE
      ITOL= ITOL + 1
      PRINT*," MAX ERROR= ",ERRM," AT X= ",XME
      PRINT*," MAX FRACTIONAL ERROR= ",FERRM," AT X= ",XMF
..
```

```
      GO TO 30
      END
      SUBROUTINE FDSM(COEF,FD,WKAREA,N,IDIM)
      DIMENSION COEF(IDIM,IDIM),FD(IDIM),WKAREA(IDIM)
      M= N-1
C     FILL FD VECTOR AS RIGHT SIDE
      DX= 1.0/N
      DO 1 J=1,M
1     FD(J)= DX*J/(N**2)
C     FILL COEF MATRIX
      DO 2 J=1,M
      DO 2 I=1,M
2     COEF(J,I)= 0.0
      DIAG = 2.0 - (1.0/(N**2))
      DO 3 J=1,M
3     COEF(J,J)= DIAG
      IF (N.EQ.2) GO TO 10
      M2= N-2
      DO 4 J=2,M2
      KM=J-1
      COEF(J,KM)= -1.0
      KP= J+1
4     COEF(J,KP)= -1.0
      COEF(1,2)= -1.0
      COEF(M,M2)= -1.0
C     PRINT RESULTS
10    PRINT*,"      "
      PRINT*,"      "
      PRINT*,"      "
      PRINT*," FINITE DIFFERENCE SOLN FOR ",M," MESH POINTS "
      PRINT*," RIGHT SIDE VECTOR= "
      PRINT*,(" RS(",J,")= ",FD(J),J=1,M)
      PRINT*," COEF MATRIX= "
      DO 12 J=1,M
12    PRINT*,(" C(",J,",",I,")= ",COEF(J,I),I=1,M)
C     CALCULATE SOLN USING THIS IMPLICIT SCHEME
      ML= 1
      IDGT= 8
      CALL LEGT1F(COEF,ML,M,IDIM,FD,IDGT,WKAREA,IER)
      PRINT*,"      "
      PRINT*," FINITE DIFFERENCE SOLN= "
      PRINT*,(" FD(",J,")= ",FD(J),J=1,M)
      PRINT*," MESH POINT LOCATIONS+ "
      DO 13 I=1,M
13    WKAREA(I)= DX*I
      PRINT*,(" X(",I,")= ",WKAREA(I),I=1,M)
      PRINT*,"      "
      RETURN
      END
      SUBROUTINE COMPFD(FD,N,IDIM)
      DIMENSION FD(IDIM)
C     COMPARE ACCURACY OF FD AT MESH POINTS
      PRINT*," ERROR OF FD AT MESH POINTS= "
      DX= 1.0/N
..
```

```
      M= N-1
      ERRM= 0.0
      FERRM= 0.0
      XME= 0.0
      XMF= 0.0
      DO 1 J=1,M
      X= DX*J
      Y= (SIN(X)/SIN(1.0)) - X
      ERR= Y- FD(J)
      FERR= ERR/Y
      IF (ABS(ERR).GT.ERRM) GO TO 2
      GO TO 5
2     ERRM= ABS(ERR)
      XME= X
5     IF (ABS(FERR).GT.FERRM) GO TO 3
      GO TO 6
3     FERRM= ABS(FERR)
      XMF= X
6     PRINT*," FOR X= ",X," ERROR= ",ERR," AND FRACTION ERROR= ",FERR
1     CONTINUE
      PRINT*," MAX ERROR= ",ERRM," AT X= ",XME
      PRINT*," MAX FRACTIONAL ERROR= ",FERRM," AT X= ",XMF
      RETURN
      END
      SUBROUTINE SOR(A,XI,XO,Y,WK,N,IDIM,ALP,CONV)
      DIMENSION A(IDIM,IDIM),XI(IDIM),Y(IDIM),WK(IDIM)
     1 ,XO(IDIM)
C
C   SUCCESSIVE OVER RELAXATION FOR AX=Y
C   XI IS AN INITIAL GUESS AND ALP IS A SOR PARAM
C
      DO 1 I=1,N
1     WK(I)= XI(I)
C   COUNT ITERATIONS FOR MIN OF 10 AND MAX OF 1000
      ITC= 1
C   ALGORYTHM FOR SOR
8     DO 20 I=1,N
      XO(I)= WK(I) + (ALP*Y(I)/A(I,I))
      DO 9 J=I,N
9     XO(I)= XO(I) - (ALP*A(I,J)*WK(J)/A(I,I))
      IF (I.EQ.1) GO TO 20
      IM1= I-1
      DO 10 J=1,IM1
10    XO(I)= XO(I) - (ALP*A(I,J)*XO(J)/A(I,I))
20    CONTINUE
C   CONVERGENCE TEST
      ICONV= 0
      DO 50 I=1,N
      IF (ABS((XO(I)-WK(I))/XO(I)).GT.CONV) ICONV= 1
      WK(I)= XO(I)
50    CONTINUE
      IF (ICONV.EQ.0) GO TO 80
      IF (ITC.GE.1000) GO TO 70
      ITC= ITC+1
..
```

```
      GO TO 8
70    PRINT*," SOR NOT CONVERGED "
      RETURN
80    IF (ITC.GT.10) GO TO 90
      ITC= ITC+1
      GO TO 8
50    PRINT*," SOR CONVERGED "
      RETURN
      END
..
```

```
      PROGRAM TIME1(INPUT,OUTPUT,)
      COMPLEX ALFA(20),Z(20,20),LAMDA(20)
      REAL A(20,20),B(20,20),WK(800),RZ(800),RALFA(40),BETA(20)
     1 ,RLAMDA(20),RRZ(20,20),FD(100,2)
      EQUIVALENCE (ALFA(1),RALFA(1)),(Z(1,1),RZ(1))
      IDIM= 20
      IDIM2= 2*IDIM
      IDIMS= IDIM*IDIM2
      IDIMF= 100
      DO 5 JJ=1,3
      T= .05 * JJ
      DO 1 ICRIT=1,2
      DO 1 N=1,20
      CALL MWRSOL(A,B,N,IDIM,ICRIT,Z,RRZ,RLAMDA,ALFA,BETA,WK,RZ,
     1 RALFA,IDIM2,IDIMS,LAMDA)
      CALL MWRCOM(RRZ,RLAMDA,IDIM,N,T)
1     CONTINUE
      DO 3 II=1,10
      N= II*10
      DX= 1.0/(N+1)
      DT= (DX**2)/2.1
      IF (DT.GE.T) GO TO 10
      IVAL= T/DT
      IVAL= IVAL+1
      DT= T/IVAL
      GO TO 30
10    DT= T
C   LOAD INITIAL VALUES
30    DO 4 I=1,N
4     FD(I,1)= 0.0
      PRINT*,"      "
      PRINT*,"      "
      PRINT*,"      "
      PRINT*," FD SOLN FOR ",N," MESH POINTS "
      DO 2 J=1,IVAL
      CALL FDSOLN(FD,AA,N,DT,WK,IDIMF)
2     CONTINUE
      PRINT*,(" FD(",I,")= ",FD(I,2),I=1,N)
      CALL COMPFD(FD,N,IDIMF,T)
3     CONTINUE
5     CONTINUE
      END
      SUBROUTINE GALT(A,B,N,IDIM)
      DIMENSION A(IDIM,IDIM),B(IDIM,IDIM)
C   FILL A AND B MATRICES
      DO 1 J=1,N
      DO 1 I=1,N
     A(J,I)= ((1.0*(I*(I-1)))/(J+I-1))
     1 - ((2.0*(I**2))/(J+I))
     2 + ((1.0*(I*(I+1)))/(J+I+1))
      B(J,I)= (2.0/(J+I+2)) - (1.0/(J+I+1))
..
```

```
     1 - (1.0/(J+I+3))
1     CONTINUE
      RETURN
      END
      SUBROUTINE COLT(A,B,N,IDIM)
      DIMENSION A(IDIM,IDIM),B(IDIM,IDIM)
C   FILL A AND B MATRIX
      DX= 1.0/(N+1)
      DO 1 J=1,N
      DO 1 I=1,N
      X= DX*J
      IF (I.EQ.1) GO TO 10
      GO TO 20
10    A(J,I)= -2.0
      GO TO 50
20    IF (I.EQ.2) GO TO 30
      GO TO 40
30    A(J,I)= 2.0-(6.0*X)
      GO TO 50
40    A(J,I)= (I*(I-1)*(X**(I-2))) - (I*(I+1)*(X**(I-1)))
50    B(J,I)= (X**(I+1)) - (X**I)
1     CONTINUE
      RETURN
      END
      SUBROUTINE MWRSOL(A,B,N,IDIM,ICRIT,Z,RRZ,RLAMDA,ALFA,BETA,
     1 WK,RZ,RALFA,IDIM2,IDIMS,LAMDA)
      COMPLEX ALFA(IDIM),Z(IDIM,IDIM),LAMDA(IDIM)
      REAL A(IDIM,IDIM),B(IDIM,IDIM),WK(IDIMS),RZ(IDIMS),RALFA(IDIM2),
     1 BETA(IDIM),RLAMDA(IDIM),RRZ(IDIM,IDIM)
C   COMPUTE A AND B MATRICES
      PRINT*,"      "
      PRINT*,"      "
      PRINT*,"      "
      PRINT*," MWR SOLUTION FOR ",N," TERM EXPANSION "
      IF (ICRIT.EQ.1) GO TO 10
      GO TO 20
10    PRINT*," COLLOCATION"
      CALL COLT(A,B,N,IDIM)
      GO TO 40
20    IF (ICRIT.EQ.2) GO TO 30
      GO TO 40
30    PRINT*," GALLERKIN"
      CALL GALT(A,B,N,IDIM)
40    PRINT*," MATRIX A= "
      DO 1 J=1,N
1     PRINT*,(" A(",J,",",I,")= ",A(J,I),I=1,N)
      PRINT*," MATRIX B= "
      DO 2 J=1,N
2     PRINT*,(" B(",J,",",I,")= ",B(J,I),I=1,N)
C   CALCULATE EIGENVALUES AND EIGENVECTORS
      IJOB= 2
      CALL EIGZF(A,IDIM,B,IDIM,N,IJOB,RALFA,BETA,RZ,IDIM,WK,IER)
      PRINT*,"      "
      PRINT*," SOLUTION VALUES "
..
```

```
      PRINT*," IER= ",IER
      PRINT*," PERFORMANCE INDEX= ",WK(1)
C     FORM COMPLEX AND REAL EIGENVALUES
      DO 3 J=1,N
      IF (BETA(J).EQ.0.0) GO TO 100
      GO TO 110
100   PRINT*," BETA(",J,")= 0.0 "
      STOP
110   LAMDA(J)= ALFA(J)/BETA(J)
      RLAMDA(J)= ALFA(J)/BETA(J)
3     CONTINUE
      PRINT*," COMPLEX EIGENVALUES= "
      PRINT*,(" LC(",I,")= ",LAMDA(I),I=1,N)
      PRINT*," REAL EIGENVALUES= "
      PRINT*,(" LR(",I,")= ",RLAMDA(I),I=1,N)
      PRINT*," COMPLEX EIGENVECTORS= "
      DO 4 J=1,N
4     PRINT*,(" EV(",J,",",I,")= ",Z(J,I),I=1,N)
C     FORM REAL EIGENVECTORS
      CALL CXNFIX(Z,RRZ,N,IDIM,RZ,IDIMS)
C     SETUP MATRICES TO CALCULATE EIGENVECTOR MULTIPLIERS
      IF (ICRIT.EQ.1) GO TO 200
      GO TO 210
200   CALL COLV(A,BETA,RRZ,N,IDIM)
      GO TO 230
210   IF (ICRIT.EQ.2) GO TO 220
      GO TO 230
220   CALL GALV(A,BETA,RRZ,N,IDIM)
C     SOLVE SYSTEM FOR MULTIPLIERS
230   ML=1
      IDGT=8
      CALL LEQT1F(A,ML,N,IDIM,BETA,IDGT,WK,IER)
      PRINT*," FOR I.C. SOLUTION IER= ",IER
      PRINT*," MULTIPLIERS ARE= "
      PRINT*,(" M(",I,")= ",BETA(I),I=1,N)
C     SCALE EIGENVECTORS BY MULTIPLIERS
      DO 6 K=1,N
      DO 6 I=1,N
6     RRZ(I,K)= RRZ(I,K)*BETA(K)
      RETURN
      END
      SUBROUTINE GALV(A,BETA,RRZ,N,IDIM)
      DIMENSION A(IDIM,IDIM),BETA(IDIM),RRZ(IDIM,IDIM)
C     FILL BETA MATRIX
      DO 1 J=1,N
1     BETA(J)= (2.0/(J+2))-(1.0/(J+1))-(1.0/(J+3))
C     FILL A MATRIX
      DO 2 J=1,N
      DO 2 K=1,N
      A(J,K)= 0.0
      DO 2 I=1,N
      A(J,K)= A(J,K)+(RRZ(I,K)*((1.0/(J+I+1))-(2.0/(J+I+2))
     1  +(1.0/(J+I+3))))
2     CONTINUE
..
```

```
      RETURN
      END
      SUBROUTINE COLV(A,BETA,RRZ,N,IDIM)
      DIMENSION A(IDIM,IDIM),BETA(IDIM),RRZ(IDIM,IDIM)
      DX= 1.0/(N+1)
C     FILL BETA MATRIX
      DO 1 J=1,N
1     BETA(J)= (DX*J)-1.0
C     FILL A MATRIX
      DO 2 J=1,N
      X= DX*J
      DO 2 K=1,N
      A(J,K)= 0.0
      DO 2 I=1,N
      A(J,K)=A(J,K)+(RRZ(I,K)*((X**I)-(X**(I+1))))
2     CONTINUE
      RETURN
      END
      SUBROUTINE SOLNT(X,T,TS,CONV)
      PIE= 3.1415926536
      TSO= 1.0-X
      M=1
1     EX= (M**2)*(PIE**2)*T
      ARG= M*PIE*X
      TS= TSO - (2.0*SIN(ARG)/(PIE*M*EXP(EX)))
      IF ((2.0/(PIE*M*EXP(EX))).LT.CONV) RETURN
3     TSO= TS
      M= M+1
      GO TO 1
      END
      SUBROUTINE MXRST(X,T,TM,RRZ,RLAMDA,IDIM,N)
      DIMENSION RRZ(IDIM,IDIM),RLAMDA(IDIM)
      TM= 1.0-X
      DO 1 I=1,N
      CT= 0.0
      DO 2 J=1,N
      IF (RLAMDA(J).LT.0.0) GO TO 2
      IF ((RLAMDA(J)*T).GT.700.0) GO TO 2
      CT= CT+(RRZ(I,J)/EXP(RLAMDA(J)*T))
2     CONTINUE
      TM= TM+(CT*((X**I)-(X**(I+1))))
1     CONTINUE
      RETURN
      END
      SUBROUTINE MXRCOM(RRZ,RLAMDA,IDIM,N,T)
      DIMENSION RRZ(IDIM,IDIM),RLAMDA(IDIM)
      CONV= 1.0E-20
C     COMPARE AT FD POINTS
      DX= 1.0/(N+1)
      ERRM= 0.0
      FERRM=0.0
      XMF=0.0
      XME=0.0
      DO 1 I=1,N
..
```

```
      X=DX*I
      CALL SOLNT(X,T,TS,CONV)
      CALL MARST(X,T,TM,RRZ,RLAMDA,IDIM,N)
      ERR= TM-TS
      FERR= ERR/TS
      IF (ABS(ERR).GT.ERRM) GO TO 10
      GO TO 20
10    ERRM= ABS(ERR)
      XME= X
20    IF (ABS(FERR).GT.FERRM) GO TO 30
      GO TO 1
30    FERRM= ABS(FERR)
      XMF= X
1     CONTINUE
      PRINT*,"     "
      PRINT*," ERROR FOR FD POINTS AT T= ",T
      PRINT*," MAX ERROR= ",ERRM," AT X= ",XME
      PRINT*," MAX FRACTIONAL ERROR= ",FERRM," AT X= ",XMF
C     COMPARE AT 99 POINTS
      DX= .01
      ERRM=0.0
      FERRM=0.0
      XMF=0.0
      XME=0.0
      DO 2 I=1,99
      X= DX*I
      CALL SOLNT(X,T,TS,CONV)
      CALL MARST(X,T,TM,RRZ,RLAMDA,IDIM,N)
      ERR= TM-TS
      FERR= ERR/TS
      IF (ABS(ERR).GT.ERRM) GO TO 50
      GO TO 60
50    ERRM= ABS(ERR)
      XME= X
60    IF (ABS(FERR).GT.FERRM) GO TO 70
      GO TO 2
70    FERRM= ABS(FERR)
      XMF= X
2     CONTINUE
      PRINT*,"     "
      PRINT*," ERROR FOR 99 POINTS AT T= ",T
      PRINT*," MAX ERROR= ",ERRM," AT X= ",XME
      PRINT*," MAX FRACTIONAL ERROR= ",FERRM," AT X= ",XMF
      RETURN
      END
      SUBROUTINE CKMFIX(Z,RRZ,N,IDIM,RZ,IDIMS)
      COMPLEX Z(IDIM,IDIM),CKI
      REAL RRZ(IDIM,IDIM),RZ(IDIMS)
C
C     STORE EIGENVECTORS AS REALS AND CHECK FOR COMPLEX
C     EIGENVECTORS (WHICH COME IN CONJUGATE PAIRS) AND
C     SET ONE EIGENVECTOR TO THE REAL PART AND THE OTHER
C     TO THE IMAGINARY PART (NOTE COMPLEX EIGENVALUES ARE
C     TAKEN AS THE REAL PART FOR APPROXIMATION)
      ..
```

```
C
      ISKIP= 0
      DO 1 J=1,N
      IF (ISKIP.EQ.1) GO TO 4
      ICK= 0
      DO 2 I=1,N
      RRZ(I,J)= Z(I,J)
      CKI= Z(I,J) - RRZ(I,J)
C     CHECK FOR COMPLEX EIGENVECTORS
      IF (CKI.NE.(0.0,0.0)) ICK=1
2     CONTINUE
      IF (ICK.EQ.0) GO TO 1
C     MAKE IMAGINARY PART NEXT EIGENVECTOR
C     AND SKIP TO FOLLOWING EIGENVECTOR
      DO 3 I=1,N
      IOFF= 2 + ((I-1+(IDIM*(J-1)))*2)
      RRZ(I,J+1)= RZ(IOFF)
3     CONTINUE
      ISKIP= 1
      GO TO 1
4     ISKIP= 0
1     CONTINUE
      PRINT*," REAL EIGENVECTORS= "
      DO 10 I=1,N
10    PRINT*,(" EV(",I,",",J,")= ",RRZ(I,J),J=1,N)
      RETURN
      END
      SUBROUTINE COMPFD(FD,N,IDIMF,T)
      DIMENSION FD(IDIMF,2)
C     COMPARE SOLUTION AT MESH POINTS
      DX= 1.0/(N+1)
      CONV= 1.0E-10
      ERRM= 0.0
      FERRM= 0.0
      XME= 0.0
      XMF= 0.0
      PRINT*,"      "
      PRINT*," FD ERROR COMPARISION AT MESH POINTS "
      DO 1 I=1,N
      X= DX*I
      CALL SOLNT(X,T,TS,CONV)
      ERR= FD(I,2) - TS
      FERR= ERR/TS
      IF (ABS(ERR).GT.ERRM) GO TO 10
      GO TO 20
10    ERRM=ABS(ERR)
      SOLMEF= ABS(FERR)
      XME= X
20    IF(ABS(FERR).GT.FERRM) GO TO 30
      GO TO 1
30    FERRM= ABS(FERR)
      SOLMFE= ABS(ERR)
      XMF=X
1     CONTINUE
..
```

```
      PRINT*,' FOR T= ',T,' MAX ERROR= ',ERRM,' AT X= ',XME
     1 ,' FRAC= ',SOLMEF
      PRINT*,' MAX FRACTIONAL ERROR= ',FERRM,' AT X= ',XMF
     1 ,' ERROR= ',SOLMFE
      RETURN
      END
      SUBROUTINE FDSOLN(FD,A,N,DT,WK,IDIMF)
      DIMENSION FD(IDIMF,2),A(IDIMF,IDIMF),WK(IDIMF)
      DX= 1.0/(N+1)
      R= DT/(DX**2)
      TMR= 1.0-(2.0*R)
C     CALCULATE NEW KNOWN FD VECTOR
      FD(1,2)= (R*(FD(2,1)+1.0)) + (TMR*FD(1,1))
      FD(N,2)= (R*FD(N-1,1)) + (TMR*FD(N,1))
      NM= N-1
      DO 2 I=2,NM
      FD(I,2)= (R*(FD(I+1,1)+FD(I-1,1))) + (TMR*FD(I,1))
2     CONTINUE
C     SET NEW SOLUTION INTO OLD SOLUTION
      DO 4 I=1,N
4     FD(I,1)= FD(I,2)
      RETURN
      END
```

AD-A100 815 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/G 20/1
INVESTIGATION OF THE NUMERICAL METHODS OF FINITE DIFFERENCES AN--ETC(U)
MAR 82 R E NAEGELI
UNCLASSIFIED AFIT/GNE/PH/81-7 NL

VITA

Robert Earl Naegeli was born on 31 January 1948 in Altadena, California. He graduated from Arroyo High School in El Monte, California in 1966. He attended Rio Hondo Junior College in Whittier, California and then San Diego State University in San Diego, California from which he received the Bachelor of Science in Physics degree in June 1970. Upon graduation he received a commission in the USAF through the ROTC program and entered active duty in May 1971. His first assignment was as a missile launch officer in the 321st Strategic Missile Wing at Grand Forks AFB, North Dakota. He broadened his operational experience by serving as a wing instructor of missile launch officers and on the staff in the Codes Division. While in the Codes Division he served as chief of the training branch and then chief of the operations branch. In August of 1979 he entered the School of Engineering, Air Force Institute of Technology.

Permanent Address: 3150 9th Avenue
Arcadia, California 91006

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: AFIT/GNE/PH/81-7	2. GOVT ACCESSION NO.: AD-A100 815
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): INVESTIGATION OF THE NUMERICAL METHODS OF FINITE DIFFERENCES AND WEIGHTED RESIDUALS FOR SOLUTION OF THE HEAT EQUATION	5. TYPE OF REPORT & PERIOD COVERED: MS Thesis
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Robert E. Naegeli, Captain, USAF	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Air Force Institute of Technology (AFIT-EN), Wright-Patterson AFB, Ohio 45433	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: Air Force Materials Laboratory (AFML/MBC), Wright-Patterson AFB, Ohio 45433	12. REPORT DATE: March, 1981
	13. NUMBER OF PAGES: 96
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Approved for public release; IAW AFR 190-17

Frederick C. Lynch, Major, USAF
Director of Public Affairs

21 MAY 1981

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Numerical Analysis
Weighting Functions
Heat Transfer
Finite Difference Theory

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The method of weighted residuals applied on the whole domain for steady state and transient heat generation problems was compared to finite difference methods. The comparison consisted of the maximum absolute error from the exact solution and computer time required for solution.

One steady state and one transient heat generation problem were solved by collocation and Galerkin weighted residual methods and finite differences. The least squares weighted residual method was also used for the steady state

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. Abstract (Continued)

problem. Both problems were one-dimensional and had Dirichlet boundary conditions. Integrals for weighted residual methods were evaluated analytically to produce recursion relations. The transient problem was solved by the reduction to ordinary differential equations method for weighted residuals.

The Galerkin method was fastest to a given accuracy for both problems evaluated. The accuracy of Galerkin and other weighted residual methods was greater than finite differences after a point at low solution accuracy. This crossover point was typically two to three digits of accuracy. The polynomial trial functions used for weighted residual solutions exhibited a numerical instability for solutions of 10 terms and over increasing the maximum absolute error. Orthogonal collocation and weighted residuals on finite elements were recommended as alternate methods.

END

DATE
FILMED

7-81

DTIC